BRL CR 243

BRL

AD

12

CONTRACT REPORT NO. 243

MCDU-8 A COMPUTER CODE FOR ONE-DIMENSIONAL BLAST WAVE PROBLEMS

ADA013864

Prepared by

Drexel University
Philadelphia, PA 19104

July 1975

USA BALLISTIC RESEARCH LABORATORIES
ABERDEEN PROVING GROUND, MARYLAND

Destroy this report when it is no longer needed.
Do not return it to the originator.

Secondary distribution of this report by originating or sponsoring activity is prohibited.

Additional copies of this report may be obtained from the National Technical Information Service, U.S. Department of Commerce, Springfield, Virginia 22151.

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

(18) BRL (19) CR-213

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
Contract Report No. 213

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (*and Subtitle*)
(6) MCDU-8-A COMPUTER CODE FOR ONE-DIMENSIONAL BLAST WAVE PROBLEMS.

5. TYPE OF REPORT & PERIOD COVERED
(9) Contract Report.

6. PERFORMING ORG. REPORT NUMBER
(14) 72-8

7. AUTHOR(s)
(10) Donald L. Tuckmantel,
James Mao
Pei Chi Chou

8. CONTRACT OR GRANT NUMBER(s)
(15) DAAD05-70-C-0175

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Drexel University
Philadelphia, PA 19104

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
USA Ballistic Research Laboratories
Aberdeen Proving Ground, MD 21005

12. REPORT DATE
(11) JUL [redacted]75

13. NUMBER OF PAGES
104

14. MONITORING AGENCY NAME & ADDRESS (*if different from Controlling Office*)

15. SECURITY CLASS. (*of this report*)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (*of this Report*)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

Blast waves
Method of characteristics
Explosive blast
Pentolite spheres

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

The method of characteristics computer code MCDU-8 which calculates solutions for one-dimensional blast wave problems is described. Since the code does not treat the detonation wave, it commences calculation when the detonation emerges from the charge surface. Subsequently, shocks are traced exactly using the Rankine-Hugoniot relations while continuous regions are calculated using the method of characteristics. Comparisons are presented between MCDU-8 solutions for spherical Pentolite and other solutions and experiments.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

We would like to acknowledge Aivars Celmins, John Kineke and Ralph Shear of the Ballistic Research Laboratories for their valuable discussions during this project. We would also like to thank Robert Ciccarelli, Robert Croman and Carmen Tanzio for their assistance in computer programming.

MCDU-8 - A COMPUTER CODE FOR THE

NUMERICAL CALCULATION OF ONE-DIMENSIONAL BLAST WAVE PROBLEMS

INTRODUCTION

The propagation of blast waves in an inviscid fluid such as air or water has always been of interest, and numerous attempts have been made to obtain its solutions. The vast majority of the methods used required that either the disturbances be weak, or that the explosions obey a given similarity constraint which is appropriate for "point source" explosions only. The only methods which can give complete solutions with prescribed initial and boundary conditions are numerical methods with the aid of digital computers.

Among the numerical calculations, finite difference techniques and the method of characteristics have been widely adopted. For the blast wave problem, shock wave propagation is one of the most important features. Therefore, the method of characteristics which allows shocks to be traced exactly is inherently more accurate than finite difference methods [1].

Although there are many research papers which use the method of characteristics to solve blast wave problems, most of them have made restrictive approximations for simplicity. Chou and Huang [2] use a constant time scheme in conjunction with the method of characteristics to solve a blast wave problem resulting from the sudden release of a highly compressed air sphere. Their computer code, MCDU-7, incorporates a strong shock approximation.

In this report, the numerical method and computer code of [2] are modified to accept any equation of state in functional form involving pressure, density, and specific internal energy. A technique for handling the reflection of the inward traveling shock from the center of the sphere is also included.

In the first section, the governing equations and their corresponding characteristic equations are presented followed by the shock equations.

In the second section the general numerical procedures as well as details concerning the calculation of certain particular points are described.

In the third section, singularities which are inherent to the blast wave problem are described first. The solution for these singularities then follows.

In the fourth and fifth sections two example problems are solved and compared to the solutions obtained from existing computer codes and to experimental results. The first problem is an expansion of a high pressure sphere with an ideal gas as the medium. The results are compared to those obtained from the characteristic code MCDU-7, which is restricted to an ideal gas medium. The second calculation is for the explosion of a spherical charge 50/50 Pentolite (50% PETN-50% TNT). The solution is compared to those obtained by; a. the Brinkley-Kirkwood theory [3], b. another computer code [4], and c. experimental data [5].

In the sixth section some conclusions which are drawn with regard to these comparisons are presented.

Appendices I and II contain an input-output description and code listing respectively.

I. GOVERNING EQUATIONS

The governing equations for one-dimensional unsteady motion of an inviscid fluid are:

conservation of mass,

$$\frac{\partial \rho}{\partial t} + \rho \frac{\partial u}{\partial r} + u \frac{\partial \rho}{\partial r} + (N-1) \rho \frac{u}{r} = 0 \tag{1}$$

conservation of momentum,

$$\rho \frac{\partial u}{\partial t} + \rho u \frac{\partial u}{\partial r} + \frac{\partial p}{\partial r} = 0 \tag{2}$$

and conservation of energy

$$\frac{\partial E}{\partial t} + u \frac{\partial E}{\partial r} - \frac{p}{\rho^2}\left(\frac{\partial \rho}{\partial t} + u \frac{\partial \rho}{\partial r}\right) = 0 \tag{3}$$

where r is the Eulerian space coordinate; t is time; u is the particle velocity; ρ is the density; p is the pressure; and E is the specific internal energy. In these equations, N is a constant, with values 1, 2, and 3, corresponding to plane, cylindrical, and spherical waves, respectively. Since most equations of state are given as a relation between p, ρ and E, we shall use this form for the equation of state in our calculations.

Equations (1), (2), and (3) are a set of first order, nonlinear, hyperbolic partial differential equations. Their characteristic directions and equations are

along

$$\frac{dr}{dt} = u + c \qquad \frac{dp}{\rho c} + du + (N-1) \frac{uc}{r} dt = 0 \tag{4}$$

along

$$\frac{dr}{dt} = u - c \qquad \frac{dp}{\rho c} - du - (N-1) \frac{uc}{r} dt = 0 \tag{5}$$

along

$$\frac{dr}{dt} = c \qquad dE - \frac{p}{\rho^2} d\rho = 0 \tag{6}$$

where the quantity c is defined as

$$c^2 = \left(\frac{\partial p}{\partial \rho}\right)_E + \frac{p}{\rho^2} \left(\frac{\partial p}{\partial E}\right)_\rho \tag{7}$$

When shocks appear in the flow field, the material properties ρ, E, p and u are discontinuous and the above mentioned partial differential equations are no longer adequate to describe the motion. A new set of equations is necessary to govern the propagation of these shock waves. These equations are

$$\rho_2(U-u_2) = \rho_1 (U-u_1) \quad (8)$$

$$p_2-p_1 = \rho_1 (U-u_1) (u_2-u_1) \quad (9)$$

$$E_2-E_1 + \frac{1}{2} (p_1+p_2) \left(\frac{1}{\rho_2} - \frac{1}{\rho_1} \right) = 0 \quad (10)$$

where U is the shock velocity, and the subscripts 1 and 2 refer to the states ahead of and behind the shock front, respectively.

II. METHOD OF CALCULATION

The numerical scheme used in this code is the constant time scheme utilizing the method of characteristics introduced by Hartree [6]. This scheme has been applied by Huang and Chou [2] for the calculation of expanding high pressure spheres and by Chen and Chou [1] for the calculation of wave propagation due to intensive in-depth energy deposition in a two-layered plate. They showed that this scheme is accurate, and can be easily adapted to computer calculations.

Although the governing equations used in the present code are different from those used in [1], the calculation procedures for the initiation of a second shock and determining the properties at a regular point in the physical plane are the same. The starting singularity is treated "exactly" by first solving for the properties across the rarefaction wave and then solving for the properties across the shock wave while matching the pressure and particle velocity at the interface (contact line). The procedures used are quite similar to those found in [2], thus the details will not be repeated here.

The arrangement of this code is quite different from previous ones. This code consists of a main control program and calculation subroutines. Each subroutine is designed to perform a specified function. These subroutines may be classified into two groups: invariant and user specified. The invariant subroutines need never be changed regardless of the physical problem or materials used. For example, the subroutines for calculating the properties at different types of points in the physical plane are common for all physical problems and materials. The subroutines that must be changed under different situations are called user specified. Each of the major subroutines and its function will be spelled out as follows:

A. Invariant Subroutines

General point subroutine (Fig. 1): Given all properties at the three points, I1, I2 and I3 along a constant time line, this subroutine calculates all properties at the point 4 on the

next time line by using the three characteristics I, II, and III.

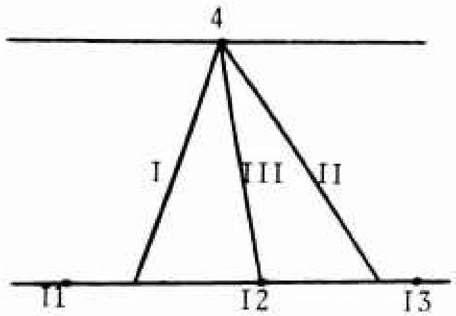


Figure 1. General Point

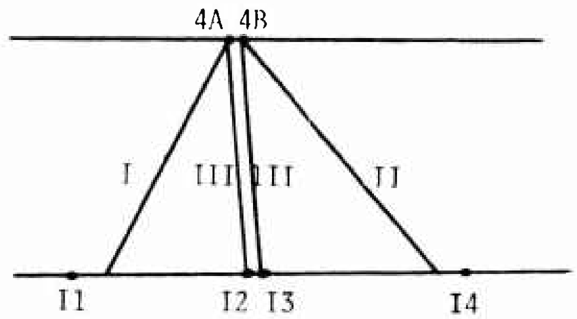


Figure 2. Interface Point

Interface (contact line) subroutine (Fig. 2): Given all properties at the interface points I2, I3 and the neighboring points I1 and I4 all on a constant time line, this subroutine calculates the new interface points 4A, 4B on the next time line, matching particle velocity and pressure at both points.

Rarefaction wave subroutine (Fig. 3): At the initial starting singularity or when a shock reaches a free surface and reflects, a rarefaction wave occurs. This subroutine calculates the properties across the rarefaction wave along a constant time line provided that the pressure behind the wave and all properties in front of the wave are given. A parameter must be specified to control the number of subdivisions that each rarefaction wave will be broken down into. By increasing the number of subdivisions we can obtain solutions as accurate as we wish.

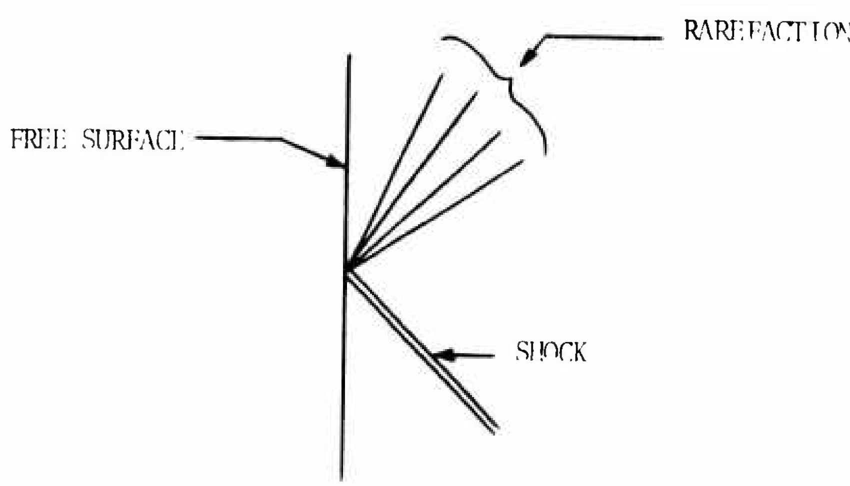


Figure 3. Shock Reflection From a Free Surface

Shock equation subroutine: The properties behind and in front of a shock must satisfy the shock equations. This subroutine calculates all properties behind a shock given all properties in front of the wave and one property behind the wave.

Shock front subroutine (Fig. 4): This subroutine calculates the properties behind the shock front at point 4B, provided all properties on the previous constant time line and one condition behind the shock front is known. The known condition behind the shock may be one of the physical variables themselves or may be given in the form of a characteristic equation. For the latter case, the characteristic grid is shown in Fig. 4A and Fig. 4B for right and left traveling shocks, respectively.

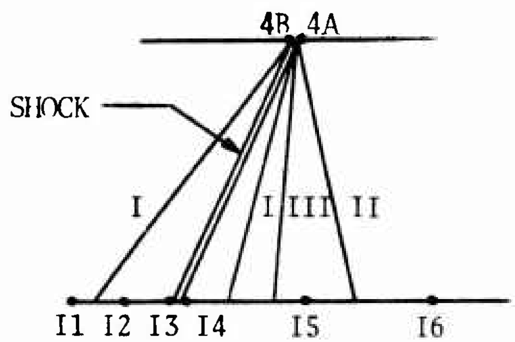


Figure 4A. Shock Front Point: Right Traveling Shock.

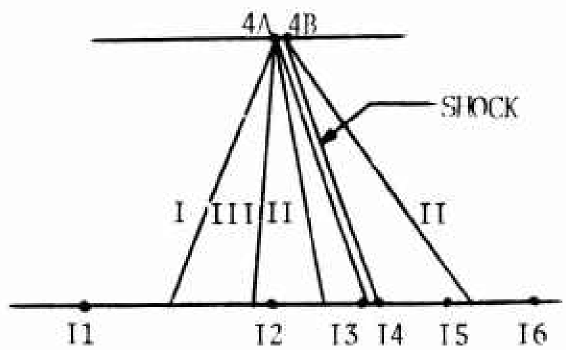


Figure 4B. Shock Front Point: Left Traveling Shock.

Rarefaction - Shock subroutine (Fig. 5): This subroutine solves for the initial singularity as well as various wave interactions. It handles two possible cases. The first case happens during the initial explosion (Fig. 5A) or during the interaction of a shock wave and a rarefaction wave (Fig. 5B). It consists of a shock traveling to the right and a rarefaction wave traveling to the left.

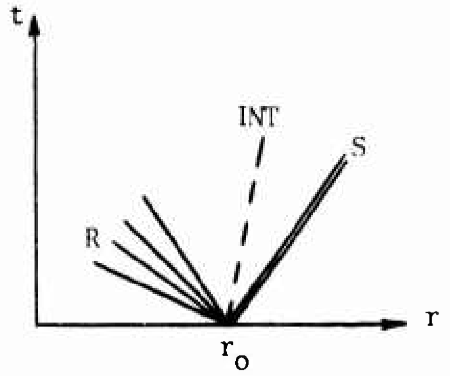


Figure 5A. Starting Singularity

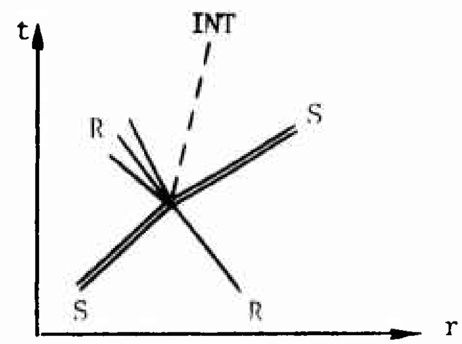


Figure 5B. Wave Interaction Point

The second case is quite similar and has a left traveling shock and a right traveling rarefaction wave as shown in Figs. 5C and 5D. An iteration procedure is used to match

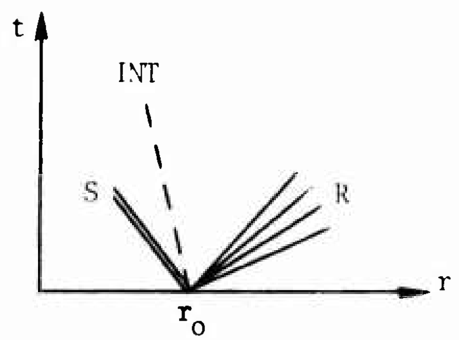


Figure 5C. Starting Singularity

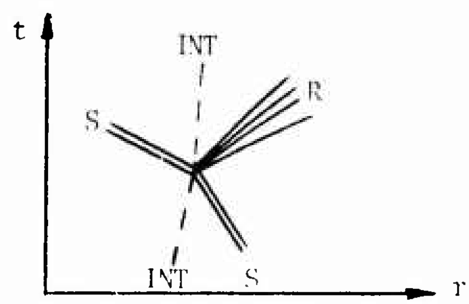


Figure 5D. Wave Interaction Point

the particle velocity and pressure at the interface (contact line).

The time increment calculation subroutine: This subroutine calculates the time increment for a new constant time line considering numerical stability and convergence. We have adopted the Courant-Friedrichs-Lewy condition as the stability criterion [7]. It was originally derived for a simple wave equation; however, it has been used very successfully for more complicated sets of equations [1], [2], and [8].

The point arrangement subroutine: After all points on a constant time line have been completely calculated, this subroutine rearranges the points before proceeding with the next time line calculations. It performs the following steps:

1. Rearranges the order of points with respect to their position.
2. Automatically adds or deletes points to maintain a relatively constant time increment.
3. Automatically deletes those points whose particle paths cross the shock waves.
4. Maintains a specified number of points in front of the main shock, avoiding the calculation of points which are not necessary.

The shock reflection subroutine: This subroutine calculates the properties at the singularity formed as a inward shock reflects from the center of symmetry. A more detailed treatment of this subroutine will be given in the next section.

The center point subroutine: This subroutine calculates the properties at the singularity occurring at the center of symmetry by the method of extrapolation.

Initial data subroutine: This subroutine assigns all properties to all points along the first constant time line and controls the subroutines used to solve the starting singularity.

B. The User Specified Subroutines

Free surface subroutine: This subroutine calculates the boundary point of a physical problem. It must be specified for different boundary conditions.

Non-dimensional subroutine: For various reasons, it is beneficial to non-dimensionalize quantities before calculation. This subroutine must be adjusted for different forms of non-dimensionalization.

Equation of state subroutine: This group of subroutines specifies the equation of state and calculates several related quantities. It consists of six subroutines; EQSTCQ, EQSTEQ, EQSTRQ, EQSTPQ, EQSTPR, and EQSTPE which calculate the sound speed, c; internal energy, E; density, ρ; pressure, p; and the derivatives $\partial p/\partial \rho$ and $\partial p/\partial E$ respectively. These subroutines must all be changed for different equations of state. Any functional or tabular relation among the density, specific internal energy and pressure could be used as the equation of state in this code.

III. SINGULARITIES

In either the expansion of a spherical compressed gas or an explosion, certain mathematical singularities must be solved before the general numerical calculations can begin. If we start calculation from the instant at which the detonation wave reaches the explosive charge surface or at the moment the highly compressed gas is released, a discontinuity of properties exists (the so-called starting singularity). We solve for this singularity by using regular characteristic methods as in [2].

At the center of symmetry, the particle velocity and radius are both zero. It can be seen from the equation of continuity (1) or from the characteristic equations (4) and (5) that the term u/r becomes uncertain at this point. This term is approximated by the derivative $\partial u/\partial r$ which is then extrapolated from the neighboring points.

In an explosion of a spherical charge there is a shock traveling towards the center of the sphere in addition to a primary strong shock propagating outward. This inward shock is usually referred to as the second shock. The existence of this second shock has been predicted by theory [9], and by numerical calculations [10], [11], and [12]. It begins as a very weak compressive discontinuity which builds up as it travels toward the center, the point of symmetry. This inward traveling shock wave will have infinite strength (becomes singular) as it collapses on the center.

The behavior of the shock near the singular point has been analytically studied by many authors. In [13] and [14] it has been shown that the relations between the change in Mach number, M, of the shock wave and a small change in the cross-sectional area, A, of the adjacent particles is given by the formula

$$\frac{\delta A}{A} = - \frac{2M\ \delta M}{(M^2 - 1)\ K(M)} \tag{11}$$

where K(M) is a slowly varying function which starts at 0.5 for a weak shock, M=1, and tends to 0.394 as $M \to \infty$ (for γ=1.4). Considering a point at a specified distance from the center of symmetry, equation (11) shows that the Mach number will be the same regardless of whether the shock is approaching or reflecting from the center. The Mach number is defined to be the ratio of current shock speed to the sound speed of the undisturbed medium. The sound speed of the undisturbed medium is the same for both reflecting and converging shocks. Therefore, the shock velocity of both the reflected and converging shock waves at any arbitrarily short distance from the center should have the same magnitude but be opposite in sign. Using this conclusion, we next present a brief description of the treatment of the shock reflection as used in this code.

Let us assume that the numerical solutions have been calculated up to a time t_1 (see Fig. 6), the time just before the shock collapses on the center. Examining Fig. 6 we can see that the converging shock intersects time line $t=t_1$, at $r=r_1$. Because r_1 is small, we assume that the shock will travel the short distance between r_1 and the center with constant speed. Therefore, the location

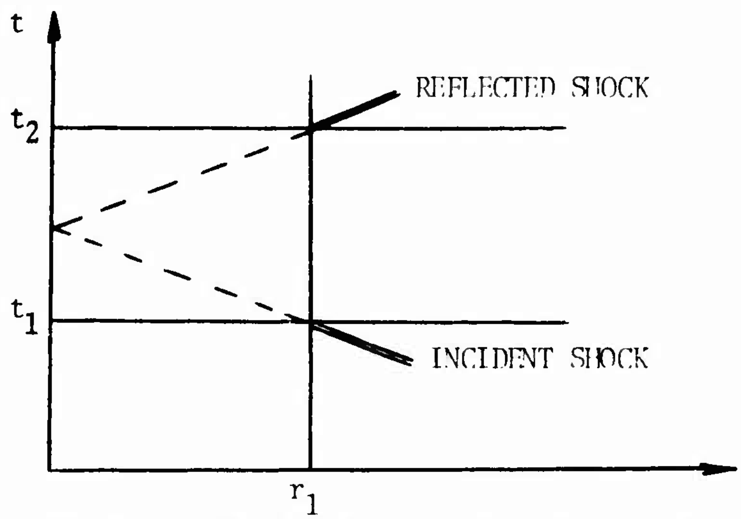


Figure 6. Shock Reflecting From Center of Cylinder or Sphere

and the velocity of the reflected shock is known assuming that the Mach number of the reflected shock is the same as the incident shock. The properties across the reflected shock can be calculated using the shock subroutine and assuming that the shock velocity is known. After solving for this singular point, the solutions of all other points can be calculated as before.

IV. A SAMPLE PROBLEM INVOLVING THE SUDDEN RELEASE OF A HIGHLY COMPRESSED AIR SPHERE

To check the accuracy of this code, the problem solved in [2] is solved by the present code and the results compared. This problem involves the sudden release of a high pressure ideal gas sphere. Before releasing, the properties in the sphere are assumed to be constant with pressure (P/P_a) and density (ρ/ρ_a) ratios with respect to the surrounding condition of 100 and 1.16, respectively. The specific heat ratio is assumed to be a constant, $\gamma=1.4$, for both media.

This same problem has been solved by the present code and the results have been compared to the ones obtained from MCDU-7, [2]. In Fig. 7, a comparison of the physical plane produced by both codes is presented. The coordinates τ and λ are the dimensionless coordinates for the time and radial distance from the center, respectively.

$$\tau = \frac{C_a t}{\varepsilon} \text{ and } \lambda = \frac{r}{\varepsilon}$$

where C_a is the constant speed of sound outside the wave zone; $\varepsilon = [E_T/(10\mathrm{x}P_a)]^{1/3}$ is length expressing energy and pressure scaling; E_T is initial total energy released; and P_a is the constant pressure outside the wave zone. For this sample problem with the initial radius of the compressed gas of 1ft (0.3048 m), the above mentioned quantities are

$$C_a = 1116.7 \text{ ft/sec } (340.37 \text{ m/sec})$$

$$P_a = 14.7 \text{ psi } (1.013 \times 10^5 \text{ N/m}^2)$$

$$\rho_a = 0.07652 \text{ lb/ft}^3 \ (1.226 \text{ kg/m}^3)$$

$$E_T = 2.19 \times 10^6 \text{ ft-lb } (2.97 \times 10^6 \text{ J})$$

The higher shock velocities produced by MCDU-7 at earlier times demonstrates the effect of the strong shock assumption used in this code. The large discrepancy between the results of both codes for the inward traveling shock is attributed to this strong shock assumption. Initially, the inward traveling shock is much weaker than the outward traveling shock; therefore, the strong shock assumption will result in larger errors for the inward shock as can be easily seen in Fig. 8. The solid line represents the results calculated by the current code and the dotted line those from MCDU-7. The coordinate π is the dimensionless pressure P/P_a. It can be seen that the strength of the inward shock, S_2,

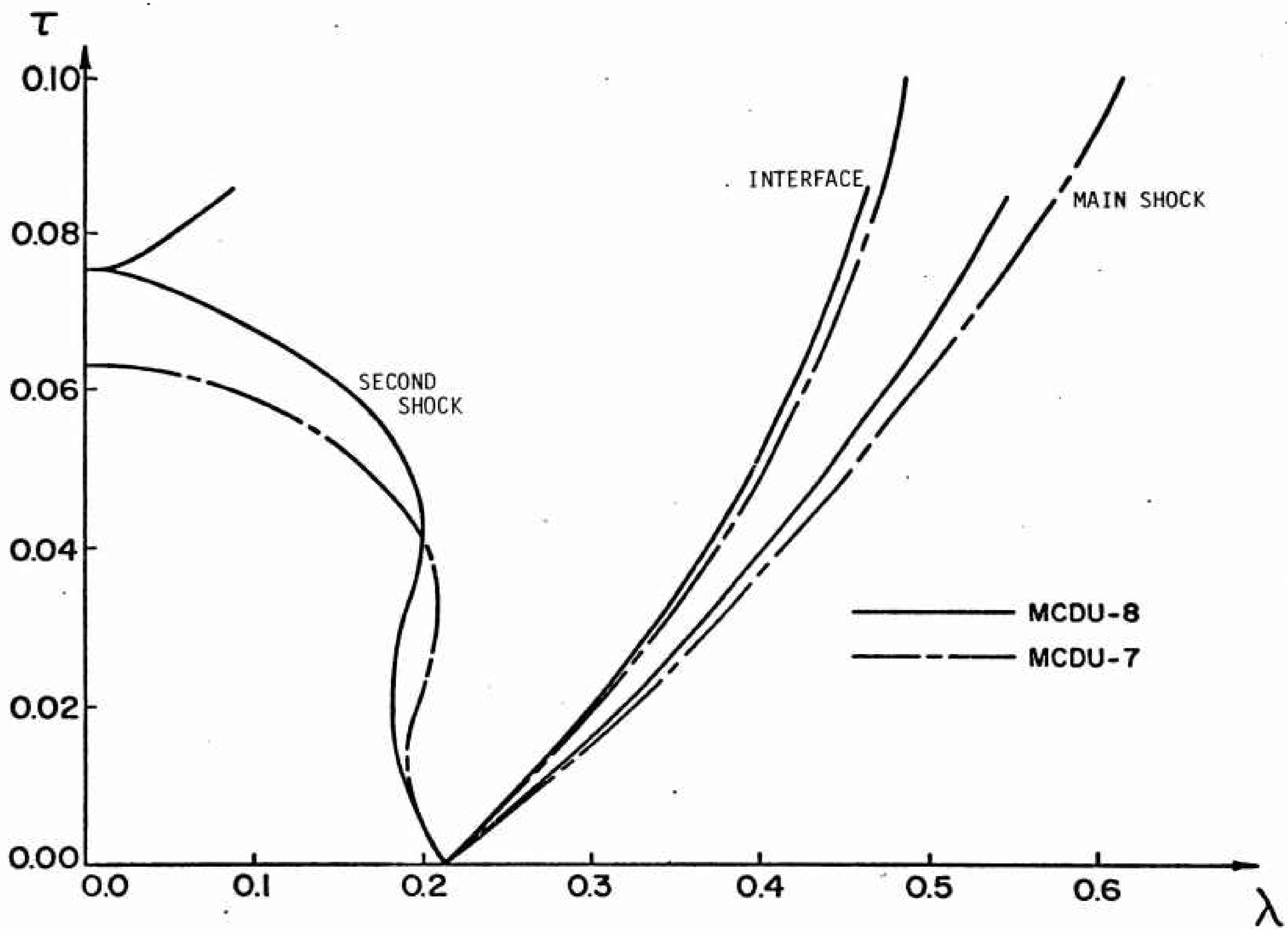


Figure 7 Plot of the Physical Plane for the Expansion of a Highly Compressed Air Sphere.

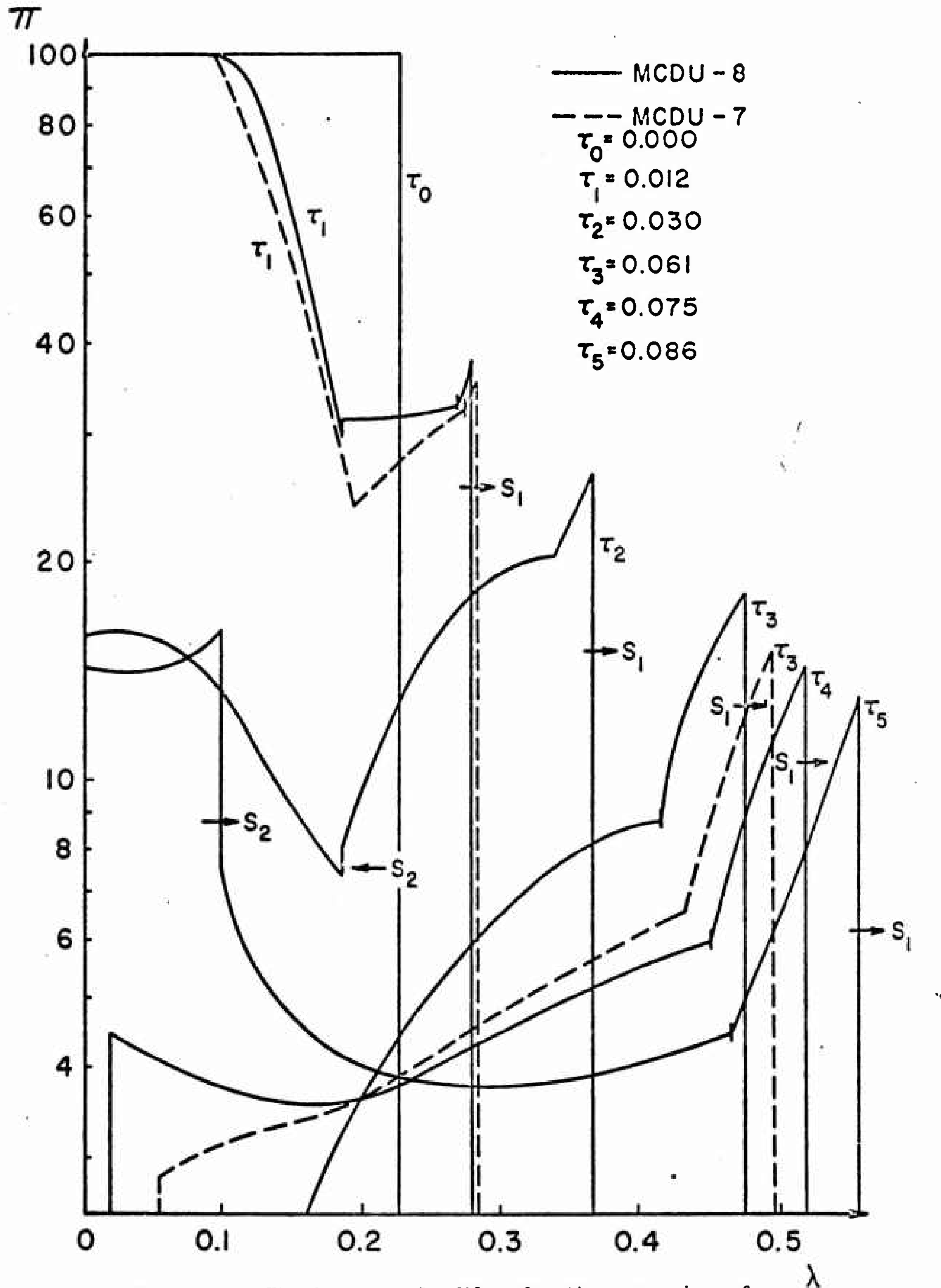


Figure 8 The Pressure Profiles for the expansion of a Highly Compressed Air Sphere.

is building while the strength of the main shock, S_1, is decaying as time increases. Between the two shocks, there is an interface where a discontinuity in the slope of the curve occurs. Just after τ=0.075, the inward shock reaches the center, reflects, and propagates outward as shown in the curve for τ=0.086. This figure also shows that at times prior to reaching the center, the wave predicted by MCDU-7 is faster than that predicted by the present code. Again, explanation can be traced to the fact that the initial pressure ratio of 100 is not high enough to justify using the strong shock approximation.

Because MCDU-7 cannot handle the singularity at the point where the inward shock reaches the center, there is no comparison of results after this time.

V. A SAMPLE PROBLEM INVOLVING A BLAST WAVE RESULTING FROM THE DETONATION OF A PENTOLITE CHARGE

The second example problem calculated by this code is a blast wave in air produced by the detonation of a spherical charge of Pentolite. The blast calculations are started at the instant the detonation wave reaches the surface of the spherical charge. We assume that the resulting gaseous products of the detonation have reached a fixed composition as we start the calculation.

The Abel equation of state [14] is used for the explosive products

$$p = \frac{N\ R\ T\ \rho}{1\ -\ a\ \rho} \qquad (12)$$

where T is temperature, R is the universal gas constant, N is the number of moles of gas per unit mass and a is the "co-volume" of the gases. For this problem, the gaseous products of the solid explosive will be characterized by an ideal gas equation of state (eq. 12 with a=0). The equation of state can be written in the familiar form $p = E\rho(\gamma-1)$. The constant specific heat ratio of the explosive gas, γ_1, is calculated from the properties at the detonation wave front at the instant it reaches the charge surface. The value of γ_1 used in this problem is 2.485. The medium surrounding the explosion is assumed to be air obeying an ideal gas equation of state with a constant specific heat ratio, γ_2, of 1.4.

The data concerning the conditions when the detonation wave front reaches the charge surface are obtained from [5]. The properties of the surrounding air are taken as standard condition at sea level, i.e., p_a = 1 atm (1.013×10^5 N/m^2), $\rho_a = 1.293 \times 10^{-3}$ gm/cm^3 (1.293 kg/m^3).

For convenience, we non-dimensionlize all variables before calculation. The dimensionless variables used for this example are listed as follows:

$$\bar{\rho} = \frac{\rho}{\rho_o}, \quad \bar{p} = \frac{p}{\rho_o c_o^2}, \quad \bar{E} = \frac{E}{c_o^2}, \quad \bar{r} = \frac{r}{r_o},$$

$$\bar{u} = \frac{u}{c_o}, \quad \bar{c} = \frac{c}{c_o}, \quad \bar{t} = \frac{t\, c_o}{r_o}$$

All variables with a "bar" on the top represent dimensionless quantities. The reference quantities used are $p_o = 2.58841 \times 10^5$ atm (2.6227×10^{10} N/m^2), and $\rho_o = 1$ gm/cm^3 (1.0×10^3 kg /m^3), $E_o = 4.217 \times 10^3$ cal/gm (1.767×10^7 J/kg) and $c_o = 8.0726 \times 10^3$ m/sec. The quantity r_o represents the charge radius.

The wave front is shown in a physical plane plot in Fig. 9. The coordinates are dimensionless time, $\bar{t}$, and radius, $\bar{r}$. A second shock with zero initial strength is initiated at the tail of the rarefaction wave very early in the calculations. Although the second shock grows in strength and propagates inward with respect to the explosive gas, the large particle velocity of the gas causes the shock to propagate away from the center when viewed with respect to a fixed coordinate system. Consequently, both shocks in Fig. 9 are propagating outward.

The relation between dimensionless velocity and the radius is presented in Fig. 10. It can be seen that during the early stages, the peak value of the particle velocity is high and concentrated within a very narrow region. This explains why in the early stages the kinetic energy is a small fraction of the total energy. As the wave propagates, the contribution of the kinetic energy increases and the potential energy or internal energy decreases. The two discontinuities in the velocity profile in Fig. 10 show the location of the two shocks. It also can be seen that the second shock propagates away from the main shock front as time increases.

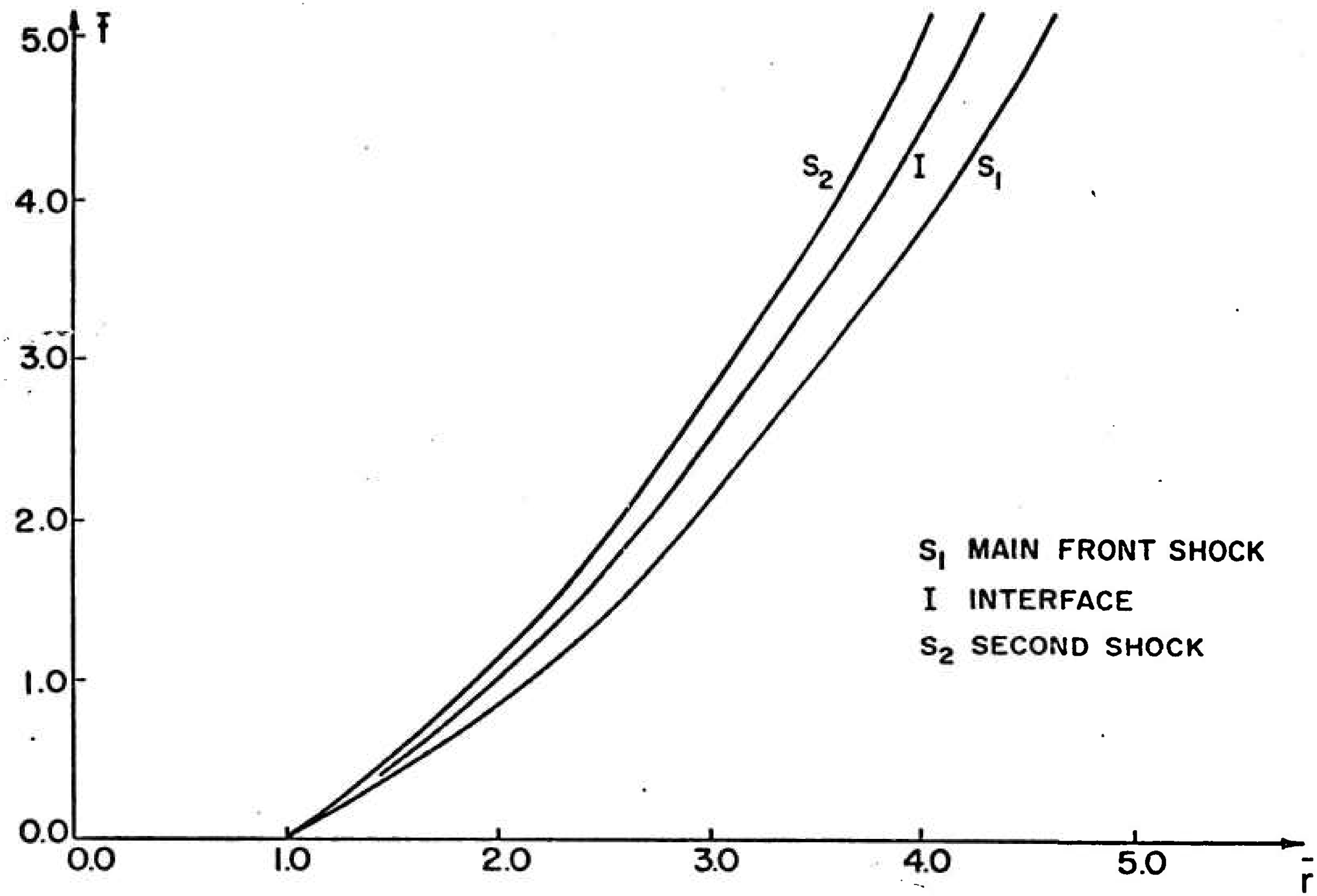


Figure 9 The shock waves in the Pentolite Blast Wave Problem.

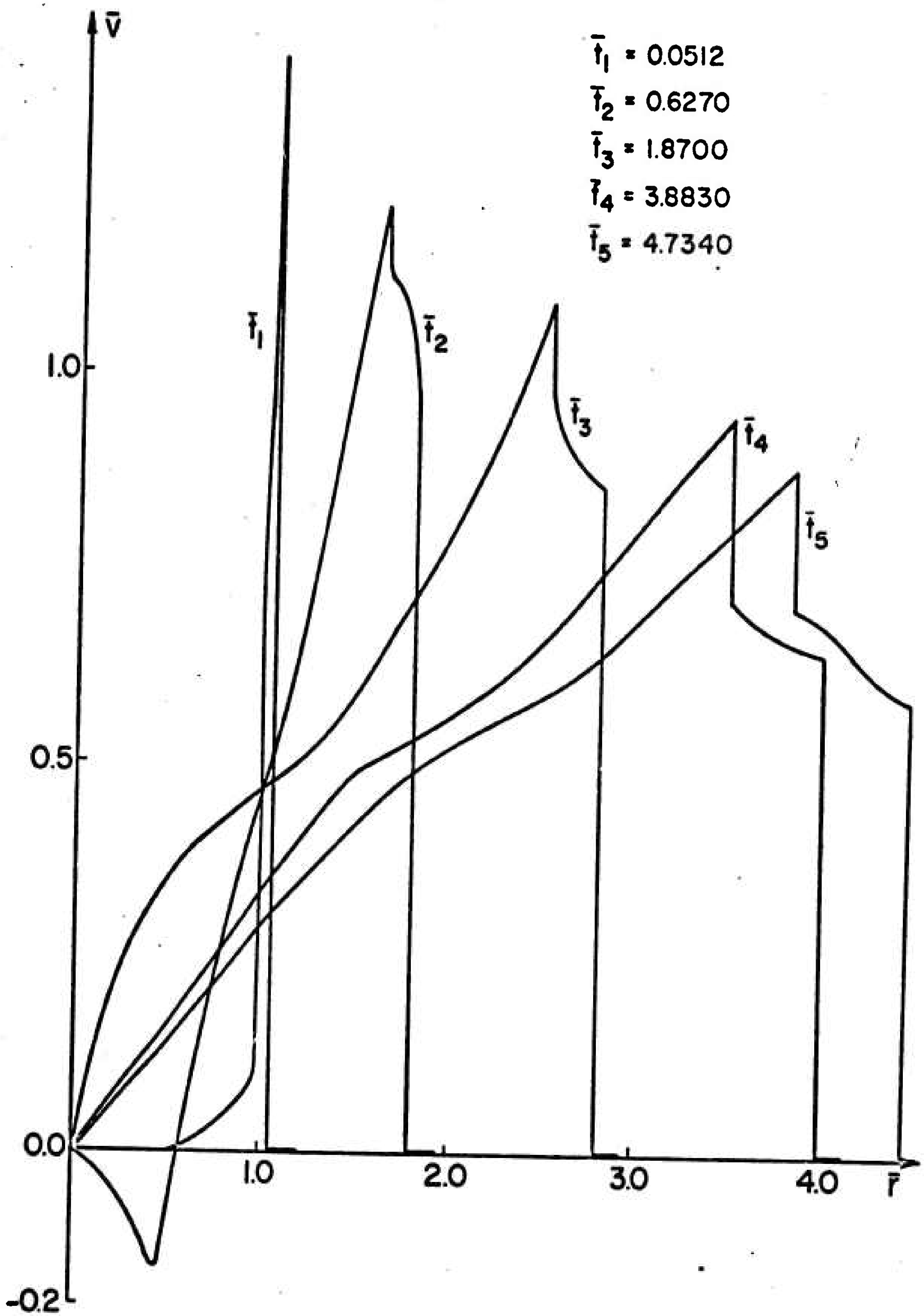


Figure 10 Velocity Profiles for the Pentolite Blast Wave Problem.

Figure 11 shows the relation between dimensionless density and radius. At any instant there are three discontinuities in the density curves. Two of them are discontinuities across the shock fronts, the third is a discontinuity across the interface.

A plot of the dimensionless pressure distribution with respect to the radius is shown in Fig. 12. (For this figure and Fig. 13, pressure has been non-dimensionalized with respect to atmospheric conditions, i.e., $P=P/P_a$). Initially, when the detonation wave reaches the charge surface, the pressure is very high; approximately a quarter million atmospheres. Immediately, a shock wave forms followed by a rarefaction wave. The pressure at the shock front drops to 720 atmospheres. In a very short time, a compression zone appears at the tail of the rarefaction wave and another shock forms (the so-called second shock or inward shock). Although the strength of the second shock is growing fast and it is traveling to the left with respect to the particle velocity in front of it, the absolute velocity of the shock carried by the explosive gas still propagates outward. As time goes on, the second shock becomes stronger while the back pressure becomes lower and it starts propagating away from the main shock front. Eventually, the inward shock will turn toward the center in the physical plane.

Finally, we compared our results to those obtained from an existing code [4], Kirkwood-Brinkley theory [3] and experimental data [5]. From Fig. 13, it is seen that for early times our code gives more favorable results in comparison to the experimental data than others. For the longer time solution, our results do not compare favorably with the experimental data. We obtained a pressure ratio across the main front shock which is higher than the experimental data. This may be the results of using a constant specific heat ratio in our calculations.

The computer code MCDU-8 has been run on both the IBM 360/75 and the Burrough B5500 computers. The first sample problem, took approximately 20 minutes on the IBM computer for all results shown in Figs. 7 and 8 with 98 points on the first time line. The second sample problem, took 80 minutes on IBM and 400 minutes on Burrough for all data shown in Figs. 9 - 13. We assigned 368 points on the first time line for this problem.

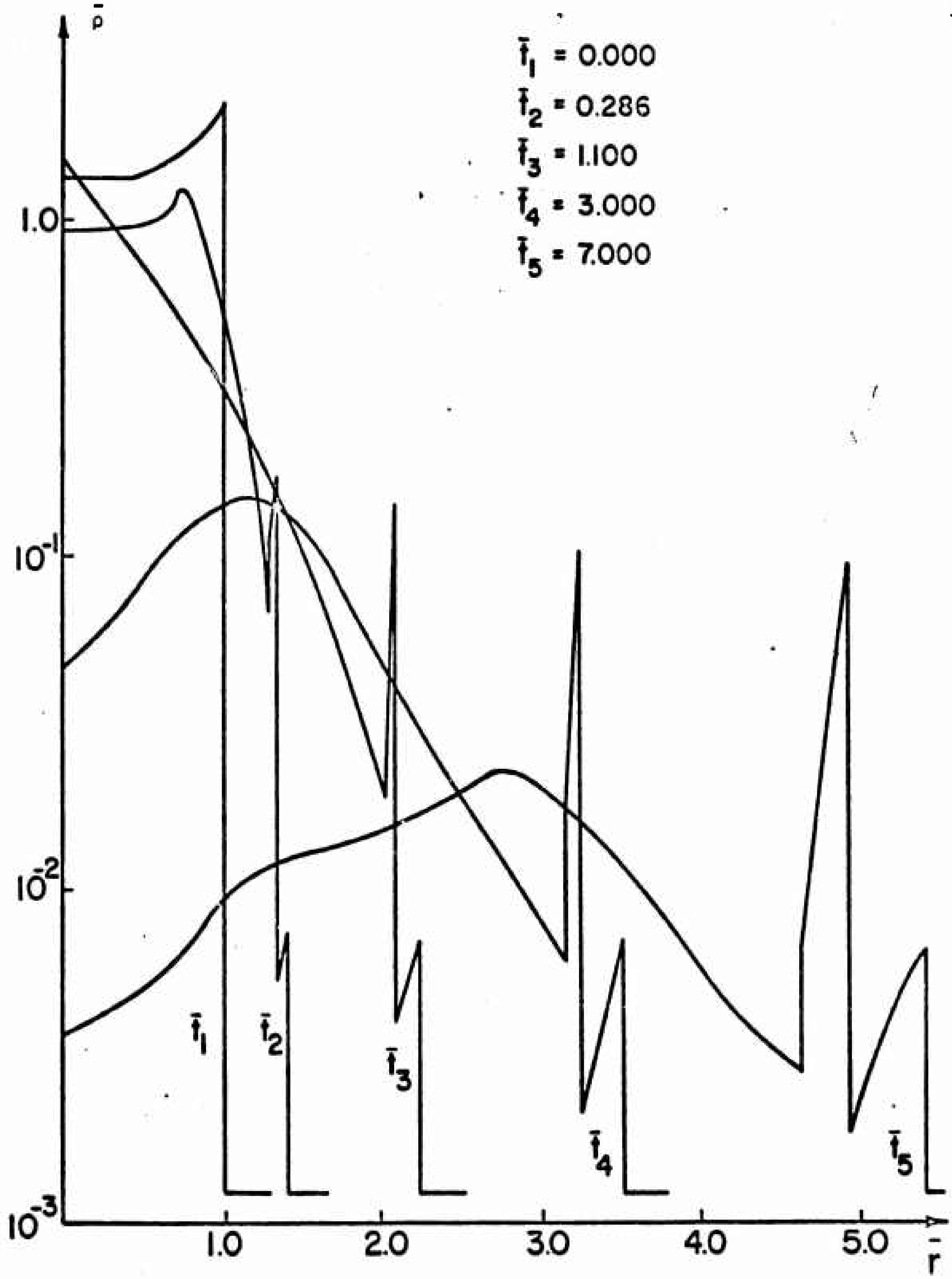


Figure 11 Density Profiles for the Pentolite Blast Wave Problem.

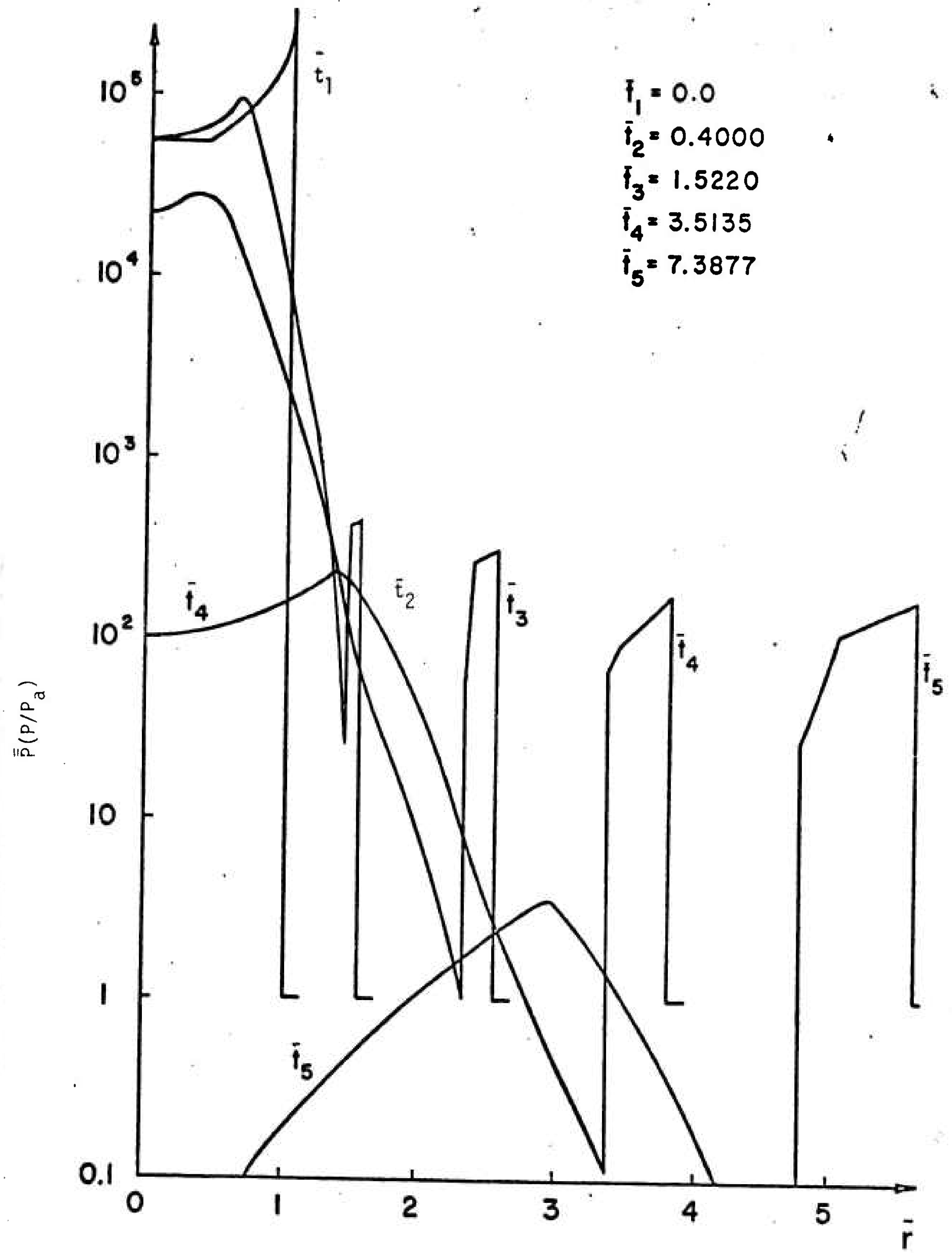


Figure 12 Pressure Profiles for the Pentolite Blast Wave Problem.

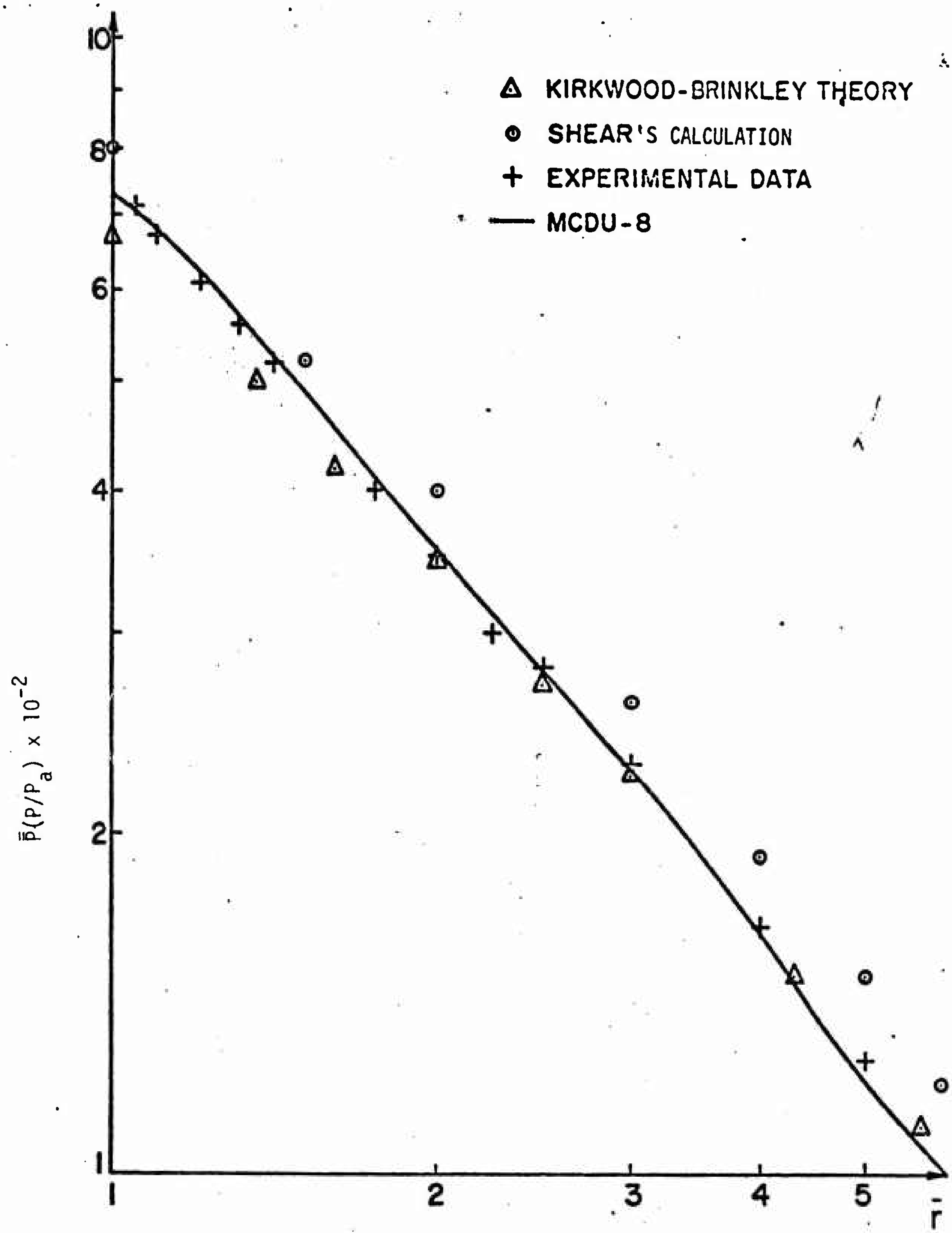


Figure 13 The comparisons of pressure ratio at the main shock front with the results obtained by other existing code, Brinkley-Kirkwood theory, and experimental data.

VI. SUMMARY AND CONCLUSIONS

A one-dimensional computer code, MCDU-8 has been developed to study the problem of a plane, spherical or cylindrical blast wave traveling through an inviscid fluid. This program uses a constant time scheme in conjunction with the method of characteristics to solve for the flow field in regions where the properties behave continuously and uses the Rankine-Hugoniot relations to treat shock waves. To demonstrate the capabilities of the code, two sample problems have been calculated.

The first sample problem treats the rapid expansion of a highly compressed (100 atmospheres) air sphere. The calculations begin when the air is released and extend past the point where the secondary shock wave reflects from the center of the sphere. The results of this calculation are compared to those of a similar characteristic code, MCDU-7 [2] which utilizes a strong shock approximation instead of the more exact Rankine-Hugoniot relations. This comparison shows that the error in peak pressure introduced by using the strong shock approximation is between 5 and 10 percent at a distance from the center of about 2.5 times the original radius of the compressed gas. Due to the limitations of MCDU-7, we were not able to compare the two codes after the secondary shock reached the center of the sphere.

The second sample problem treats the flow field produced by the detonation of a spherical charge of Pentolite. Like the previous problem, the calculations begin when the detonation wave reaches the surface of the explosive. The results of this calculation are compared to those obtained from Kirkwood-Brinkley theory [3], experimental data [5], and a computer code developed at BRL [4]. This comparison shows that during the early stages of computation, MCDU-8 produces results that are in closer agreement to the experimental data than the two other means of calculation. It has been concluded that if one is only interested in the main shock front, then the Kirkwood-Brinkley theory is adequate. However, if detailed information concerning the entire flow field is of interest then, the present code will give a more complete analysis than other available methods.

It is hoped that in the future we will be able to extend MCDU-8 to include the actual detonation calculations thus handling the complete explosion problem.

VII. REFERENCES

1. Chen, S. and Chou, P.C., "A Method of Characteristic Code for Energy Deposition Calculations (MCDIT-3)", BRL Contract Report No. 36, 1971.

2. Huang, S.L. and Chou, P.C., "Calculation of Expanding Shock Waves and Late Stage Equivalence", D.I.T. Report No. 125-12, 1968.

3. Shear, R.E. and Wright, E.Q., "Calculated Peak Pressure-Distance Curves for Pentolite and TNT", Ballistic Research Laboratories Memorandum Report No. 1423, Aberdeen Proving Ground, Maryland, 1962. (AD #287258)

4. Private Communication with Mr. Ralph E. Shear, Ballistic Research Laboratories, Aberdeen Proving Ground, Maryland.

5. Goodman, H.J., "Compiled Free-Air Blast Data on Bare Spherical Pentolite", Ballistic Research Laboratories Report No. 1092, Aberdeen Proving Ground, Maryland, 1960. (AD #235278)

6. Hartree, D.R., <u>Numerical Analysis</u>, Oxford University Press, 1958.

7. Issacson, E. and Keller, H.B., <u>Analysis of Numerical Methods</u>, John Wiley & Sons, Inc., pp. 489, 1966.

8. Tuckmantel, D.L., "A Numerical Solution of the One-Dimensional Plate Slap Problem Using the Method of Characteristics", Doctoral Thesis, Drexel University, 1971.

9. Whitman, G.E., "The Propagation of Spherical Blast", Proc. Roy. Soc. (London), 1950, pp. 571-581.

10. Berger, S.A. and Holt, M., "Implosive Phase of a Spherical Explosion in Sea Water", The Physics of Fluids, Vol. 5, No. 4, pp. 426-431, April, 1962.

11. Berry, F.J., Butler, D.S. and Holt, M., "The Early Development of Spherical Blast From a Particle Charge", Proc. Roy. Soc. (London), A227, 258, 1955.

12. Wecken, Von Fr., "Expansion Einer Gaskugel Hohen Druckes", Z. Angew Math. Mech., 1950.

13. Chisnell, R.R., "The Motion of a Shock Wave in a Channel, with Applications to Cylindrical and Spherical Shock Waves", Journal of Fluid Mechanics, 2, pp. 286-298, 1957.

14. Whitam, G.B., "On the Propagation of Shock Waves Through Regions of Non-Uniform Area or Flow", Journal of Fluid Mechanics, 4, pp. 337-360, 1958.

15. Duvall, R.E., "Applications", Chapter 9, in Dynamic Response of Materials to Intensive Impulsive Loading, Edited by P.C. Chou and A. K. Hopkins, published by Air Force Materials Laboratory 1972.

APPENDIX I - COMPUTER CODE DESCRIPTION

A. GENERAL DESCRIPTION OF MCDU-8

MCDU-8 is written in single precision FORTRAN IV. It is designed to numerically solve the equations governing a spherical blast wave, by using a constant time step iteration scheme in conjunction with the method of characteristics.

The blast wave problem consists of a sphere of highly compressed gas (designated region one) surrounded by another gaseous area (designated region two) which is relatively lower in pressure.

MCDU-8 input data begins with a card, termed the 'option' card, on which the user selects the job options applicable to the type of problem the user wishes to run. MCDU-8 is best utilized by running a problem to completion in a series of small computer runs rather than all at once. The option card provides an easy method for doing this. After reviewing the output from any particular computer run, the user has three options:

1) The problem may be carried out to a larger time using the same time step as was used in the previous run;
2) the problem may be carried out to a larger time using a different time step or,
3) the same run may be repeated using a different time step.

The first two options give the user control over the rapidity with which MCDU-8 calculates a solution while the third option allows the user to improve the accuracy of any particular run.

B. INSTALLATION DEPENDENT FEATURES

MCDU-8 utilizes two data files which must be made available to the program. The opening of data files and the devices on which they are stored, for example, tape and disk, is a function of the job control language and the facilities available at any particular installation. The user should insure that the two files have the following characteristics:

1) The files must have the unit numbers one (1) and two (2);
2) they must have a physical record size of at least five (5) words;
3) a minimum record length of 2010 records (10,050 words) and,
4) all I/O is unformated and performed serially (random access is NOT used).

C. OPTION CARD

MCDU-8 always requires at least one card of input, termed the 'option' card, on which the user specifies what actions the program is to take in solving a problem. This card is always the first card of the input deck. The six variables initialized by this card are listed in Table 1 for reference. Examples of the option card's uses are given in later sections.

TABLE 1 OPTION CARD VARIABLES

VARIABLE	COLUMNS	FORMAT	DESCRIPTION
ISTART	1-2	I2	= -1 (Continue the problem with card input). = 0 (Start a new problem with card input). = 1 (Continue the problem with file one input). = 2 (Rerun previous run with file two input).
IPUNCH	3	I1	= 0 (No punched output). = 1 (Punch out the last calculated time line. These cards are used with ISTART =-1 to continue a problem).
IDUMP	4-6	I3	(Calculated time lines are periodically stored on file one for safe keeping. I3 specifies how many lines are to be calculated before a line is dumped onto file one).
IDT	7	I1	= 0 (No new time step). = 1 (New time step is to be specified in columns 8 through 22).
DT2	8-22	E15.8	(New time step to be used if IDT=1).
TMAX	23-37	E15.8	(Problem time to which MCDU-8 is to calculate a solution).

D. DEFINING A NEW PROBLEM

Figure 14 illustrates the cards needed to define a new problem. Besides the option card, six additional input cards are required.

The first card in Fig. 14 is the option card. A new problem is signaled by setting ISTART=0. IPUNCH has been set equal to one. Upon completion of a computer run, the final time line will be punched out. The user need only include an option card at the beginning of these punched cards with ISTART=-1 in order to continue a run. IDUMP has a value of three, thus every third time line calculated will be dumped onto file one. When an option card is encountered with ISTART=1, the last time line dumped onto file one will be read and used to continue a problem.

When a problem is first defined, a singularity exists at the interface between regions one and two because of the difference in pressure. MCDU-8 has the ability to pick its own time step. The user may specify a time step by setting IDT=1 and placing the time step in columns 8 through 22. Accidently specifying too large a time step may cause serious difficulties with the program logic. As in example 2 , it is usually good policy to let MCDU-8 calculate the time step and to run the problem out several time lines until the rarefaction wave occurring at the singularity becomes "smeared" through out region one. This is done by setting IDT=0. Notice that TMAX has been set equal to fifty microseconds.

Card two contains the initial time step to be used to calculate the properties about the singularity. This consists of a shock travelling into the region two and a rarefaction wave with velocity towards the center of region one. For good results, a time step should be chosen so that after the calculation of the initial singularity about the interface, the head of the rarefaction wave has traveled approximately three percent or less of the distance from the interface to the center of region one. In this region, the shock strength is assumed constant and properties remain constant along characteristic directions. Following these assumptions, the code assigns mesh points at locations where the characteristics eminating from the singularity cross the first time line. The code then uses stability criterion to determine the time increment to the next time line. Format is E15.8.

Card three supplies the specific heat ratios (GAMMA) of the two regions. In the example given, these values are both 1.4. Format is 2E15.8.

Card four lists the dimensions of the data supplied to the program. This is only a programming convenience for the user and will label the output for documentation purposes. If this card is left blank, no labeling will result (see Table 2).

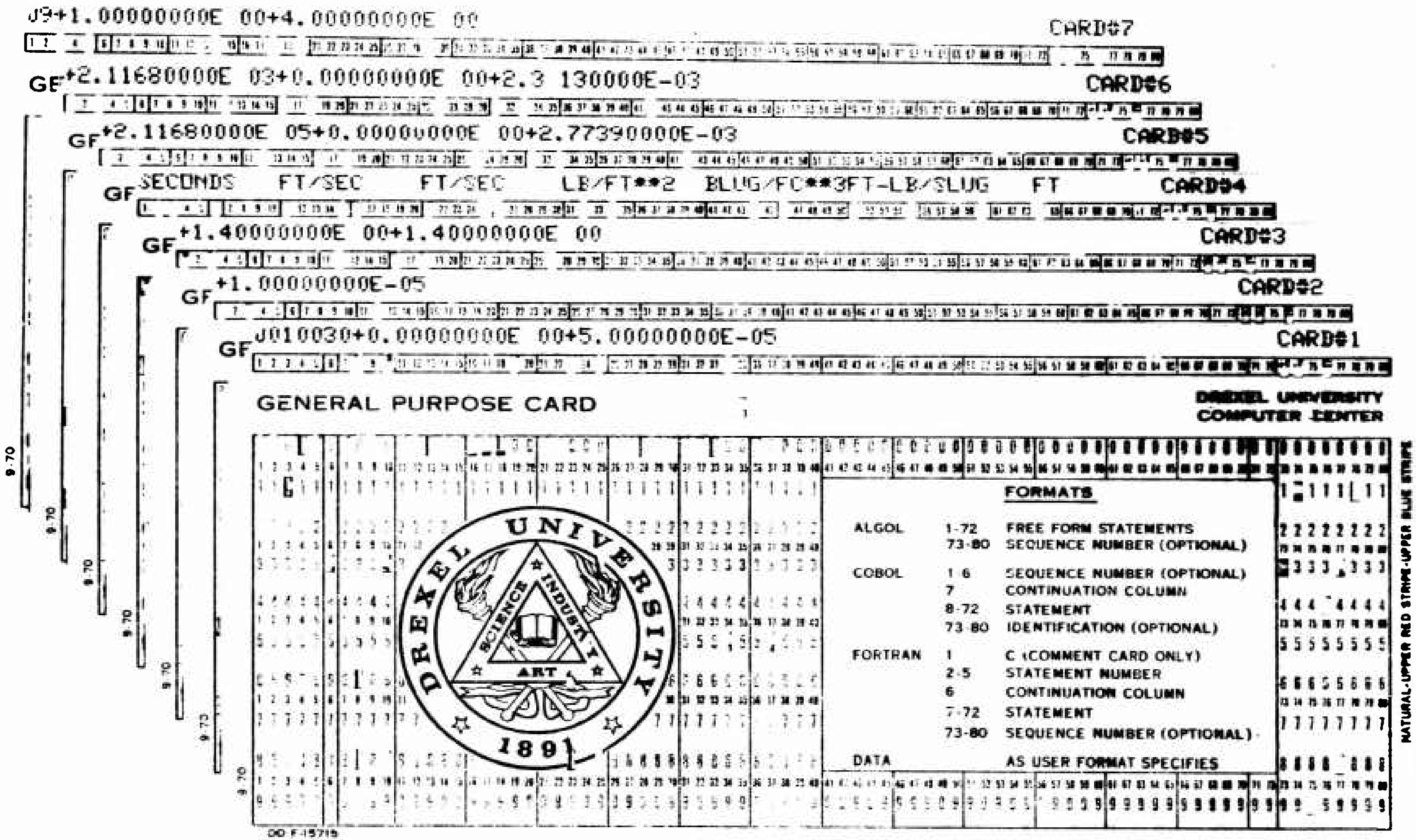


Figure 14. Sample Data for Starting a New Problem.

TABLE 2 - CARD 4 - OUTPUT LABEL

QUANTITY	EXAMPLE	COLUMNS
Time	Seconds	1-10
Particle Velocity	Ft/Sec	11-20
Sound Speed	Ft/Sec	21-30
Pressure	Lb/Ft**2	31-40
Mass Density	Slug/Ft**3	41-50
Specific Energy	Ft-Lb/Slug	51-60
Length	Ft	61-70

Card five defines the physical properties or region one: pressure (PI1), particle velocity (UI1) and density (RHI1). In the example of Fig. 14, pressure is equal to 2.1168 10^5 lb/ft^2, particle velocity is zero, and density is 2.7739 10^{-3} $slugs/ft^3$. Format is 3E15.8.

Card six contains the physical properties of region two. Pressure (PI2) in Fig. 14 is 2.1168 10^3 lb/ft^2, particle velocity (UI2) is zero, and density (RHI2) is 2.3913 10^{-3} $slugs/ft^3$. Format is 3E15.8.

Card seven contains three values. The first value supplied on this card, (IN), specifies the number of subdivisions the initial rarefaction wave is to be divided into for calculation purposes. In the example of Fig. 14, IN is set equal to nine. Ideally, the distribution of pressures from the first point in the rarefaction wave to the last point should be smooth and free from large jumps. Only experience can determine what value of IN will give a reasonable solution to a problem. Theoretically, we can make the solution as accurate as we wish by choosing large values of IN. Practically, IN should never exceed twenty-nine as MCDU-8 can only handle up to this many subdivisions without enlarging the storage allocation. Format is I2.

The second value supplied on this card, (XZ), is the initial radius of the sphere. Format E15.8.

The third value supplied on this card is the pressure jump tolerance, (PTOL). PTOL specifies what fractional percent pressure rise per unit distance must be present between two points before a shock wave will be initiated. MCDU-8 has the capability to initiate one left traveling shock wave. In Fig. 14, this has a value of 4 which is equivalent to a 4000 percent pressure jump per foot. This value appears to yield satisfactory results for this problem. This value may need to be adjusted to suit a particular problem. Format is E15.8.

E. RESTARTING FROM FILE ONE

The IDUMP variable on the option card specifies how many time lines are to be calculated between dumps on file one. If IDUMP is set equal to zero, no time lines will be dumped. Each new time line dumped on file one replaces the previous one. If after a run, the user decides to carry calculations out to a greater time, the user need only to read in an option card with ISTART=1. The last line dumped will be read off of file one and calculations will proceed from there.

By setting IDT=1, the user may also change the time step by supplying the new time step in columns eight through twenty-two on the option card. In the example of Fig. 15, a restart fro file one has been called for (ISTART=1), no punched output is requested (IPUNCH=0), every time line will be dumped (IDUMP=1), a new time step is called for (IDT=1, DT2=1.0 10^{-6} seconds), and the problem will be calculated out to a time of 100 microseconds (TMAX: 100.0 10^{-6} seconds).

An added feature of the file one restart is if a job terminates abnormally (such as running out of computer time) and the program was not executing I/O with file one (causing parts of two different time lines to be saved) the user may restart the problem as explained above and MCDU-8 will proceed from the point of termination.

F. RESTARTING FROM FILE TWO

Whenever a new job is begun, the very first time line calculated is dumped onto file two. A user may repeat the same run over as many times as desired by simply reading in an option card with ISTART=2. (See Fig. 16) by changing the time step and comparing the results with earlier runs, the convergence of the problem to an accurate solution can be checked. In the example of Fig. 16, the time step has been set to one-half microsecond.

Remember that whenever a new run is started, the first time line of the run replaces the first time line of the old run.

G. RESTARTING FROM PUNCHED OUTPUT

If several problems must be run on MCDU-i concurrently, it becomes impractical to provide disk or tape restart capabilities for every problem. MCDU-8 can remember time lines for only one problem at a time. By setting IPUNCH=1 on the option card, the last time line calculated at the end of a run will be punched out. One need only place an option card with ISTART=1 at the beginning of this punched output deck, being careful to preserve the order of the cards, in order to continue calculations from the end of the previous run of that problem. All of the other features of the option card are still available to the user of MCDU-8. (See Fig. 17).

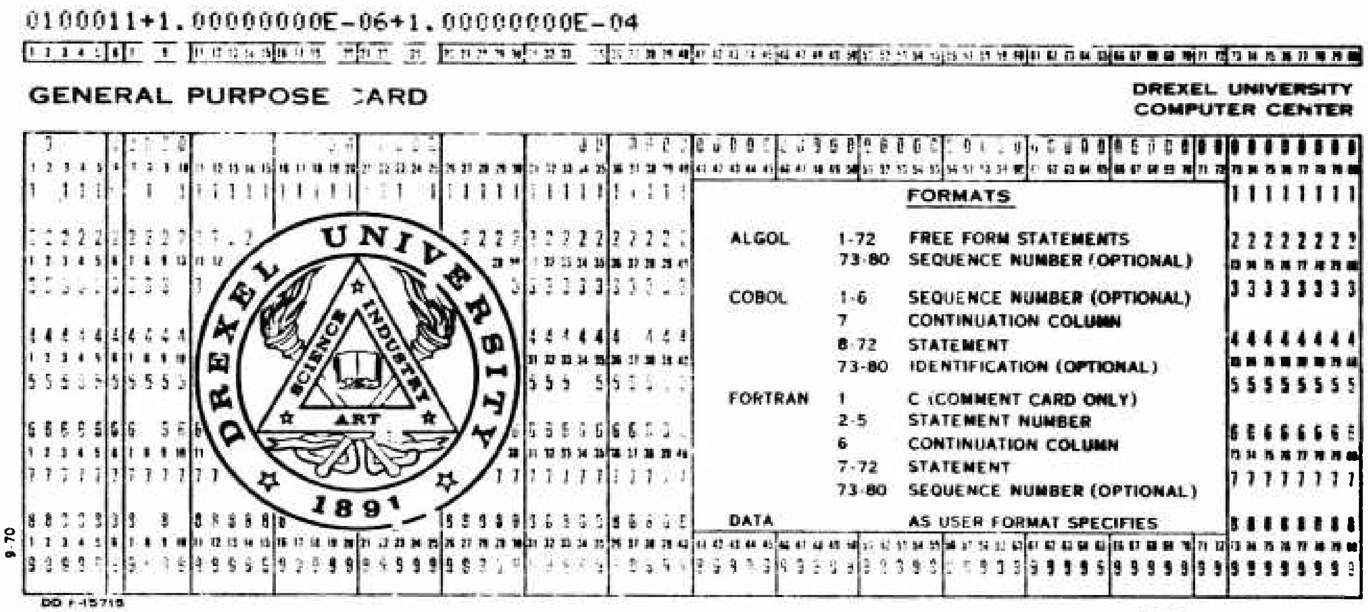


Figure 15. Sample Data for Restarting From File #1.

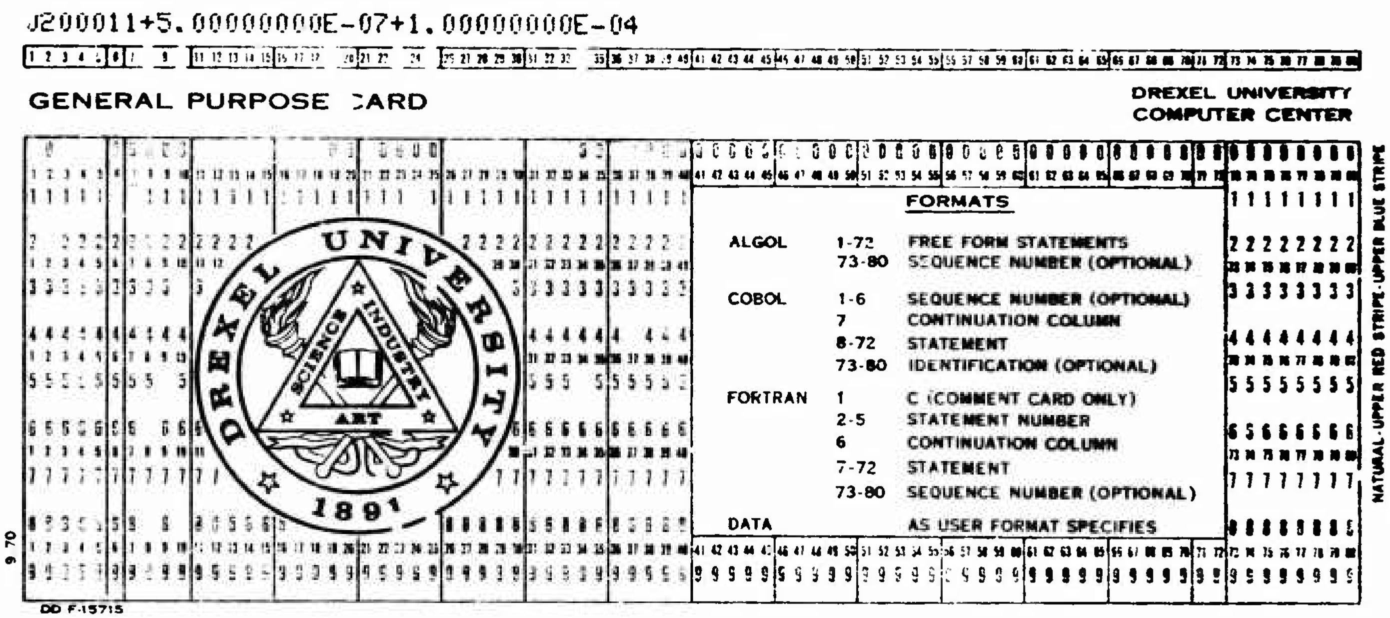


Figure 16. Sample Data for Restarting From File #2.

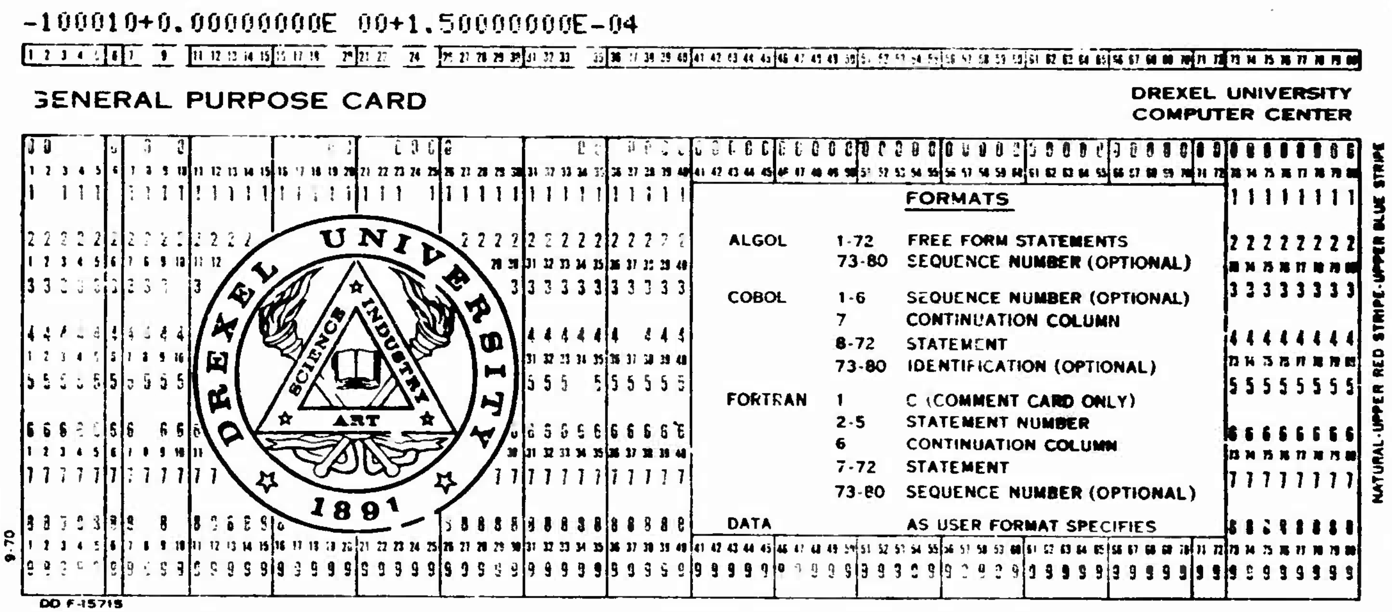
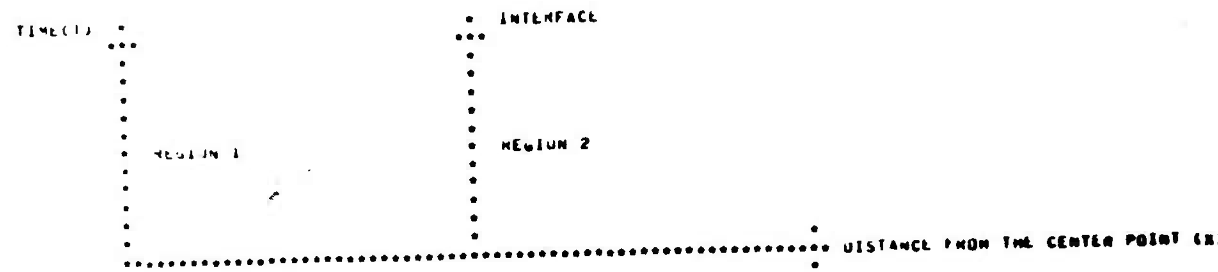


Figure 17. Sample Data for Restarting From Punched Output.

H. SAMPLE OUTPUT

Figures 18 thru 23 present sample output for MCDU-8. The information is completely labeled and should be self explanatory.



TITLE ABBREVIATIONS

X =DISTANCE FROM THE CENTER POINT
U =PARTICLE VELOCITY
C =SOUND SPEED
P =PRESSURE
RR=MATERIAL DENSITY
E =SPECIFIC ENERGY

(ISTART) 0
(IPUNCH) 1
(IDUMP) 3
(IDT) 0
(DT) .0
(TMAX) 5.000000000E-05

Figure 18. Option Card Printout.

```
DIMENSIONAL INPUT DATA


TIME UNITS ARE(SECONDS    )
INITIAL TIME= 0.0
(T)TIME OF THE FIRST TIME LINE          1.00000000E-05
(TMAX)MAXIMUM NOW TIME                  5.00000000E-05

GAMMA FOR REGION ONE 1.40000000E 00
GAMMA FOR REGION TWO 1.40000000E 00

  (FT/SEC     )  (FT/SEC     )  (LB/FT**2  )  (SLUG/FT**3)  (FT-LB/SLUG)
REGION ONE PROPERTIES
      UI1            CI1              PI1              RHI1              EI1
  .0           1.03361435E 04 2.11680000E 05 2.77390000E-03 1.90778327E 08
REGION TWO PROPERTIES
      UI2            CI2              PI2              RHI2              EI2
  .0           1.11323512E 03 2.11680000E 03 2.39130000E-03 2.21302221E 06

(IN)THE INITIAL RAREFACTION IS DIVIDED INTO     9 SUBDIVISIONS
(XZ)RADIUS OF REGION ONE  1.00000000E 00(  FT       )

(PTOL)SHOCK PRESSURE JUMP TOLERANCE  4.00000000E 00



NON-DIMENSIONAL INPUT DATA


INITIAL TIME= 0.0
(T)TIME OF THE FIRST TIME LINE          1.03361435E-01
(TMAX)MAXIMUM NOW TIME                  5.16807176E-01
REGION ONE PROPERTIES
      UI1            CI1              PI1              RHI1              EI1
  .0           1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
REGION TWO PROPERTIES
      UI2            CI2              PI2              RHI2              EI2
  .0           1.07703141E-01 7.14285714E-03 8.62071452E-01 2.07142260E-02

(XZ)RADIUS OF REGION ONE  1.00000000E 00

(PTOL)SHOCK PRESSURE JUMP TOLERANCE  4.00000000E 00


CONVERSION FACTORS BACK TO DIMENSIONAL QUANTITIES

QUANTITY MULTIPLY BY
X          1.00000000E 00
T          9.67478620E-05
U          1.03361435E 04
C          1.03361435E 04
P          2.96352000E 05
RH         2.77390000E-03
E          1.06835863E 08
```

Figure 1-. Dimensional and Non-Dimensional Input Data Printout.

BEST AVAILABLE COPY

THESE POINTS DEFINE THE INITIAL SINGULARITY ABOUT THE INTERFACE

POINT NO.	X	U	C	P	RH	E
RAREFACTION IN REGION ONE						
1	1.00000000E 00	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
2	1.00000000E 00	4.50526819E-02	9.90079648E-01	6.70428152E-01	9.95712759E-01	1.75264404E 00
3	1.00000000E 00	9.27066746E-02	9.81445773E-01	6.26562590E-01	9.10628223E-01	1.72066394E 00
4	1.00000000E 00	1.43308968E-01	9.71321474E-01	5.82701028E-01	8.64628101E-01	1.68475965E 00
5	1.00000000E 00	1.97285464E-01	9.60521286E-01	5.38839465E-01	8.17629320E-01	1.64750204E 00
6	1.00000000E 00	2.55166217E-01	9.48939006E-01	4.94977903E-01	7.69523503E-01	1.60000962E 00
7	1.00000000E 00	3.17618175E-01	9.36440723E-01	4.51116341E-01	7.20182762E-01	1.56503076E 00
8	1.00000000E 00	3.85504457E-01	9.22844524E-01	4.07254779E-01	6.69421121E-01	1.52070931E 00
9	1.00000000E 00	4.59966695E-01	9.07939635E-01	3.63393217E-01	6.17110535E-01	1.47200139E 00
10	1.00000000E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41092795E 00
INTERFACE						
11	1.00000000E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41092795E 00
12	1.00000000E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
SHOCK IN REGION TWO						
13	1.00000000E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
14	1.00000000E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142266E-02

SHOCK VELOCITY 6.60149392E-01

Figure 20. Initial Singularity Printout Generated by A New Problem.

```
MCDUB.TIME LINE     1 MCDUB.TIME LINE     1 MCDUB.TIME LINE   1 MCDUB.TIME LINE   1 MCDUB.TIME LINE   1 MCDUB.TIME LINE   1
TIME OF THIS LINE                         1.03361435E-01
TIME INCREMENT TO PREVIOUS TIME LINE      1.03361435E-01
LINE REGION POINT         X               U              C               P               RH               E
   1      1     1  .0                .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     2  7.86525057E-03    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     3  1.57305011E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     4  2.35957517E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     5  3.14610023E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     6  3.93262528E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     7  4.71915034E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     8  5.50567540E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1     9  6.29220045E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    10  7.07872551E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    11  7.86525057E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    12  8.65177563E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    13  9.43830068E-02    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    14  1.02248257E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    15  1.10113508E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    16  1.17978759E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    17  1.25844009E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    18  1.33709260E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    19  1.41574510E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    20  1.49439761E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    21  1.57305011E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    22  1.65170262E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    23  1.73035513E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    24  1.80900763E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    25  1.88766014E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    26  1.96631264E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    27  2.04496515E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    28  2.12361765E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    29  2.20227016E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    30  2.28092266E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    31  2.35957517E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    32  2.43822768E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    33  2.51688018E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    34  2.59553269E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    35  2.67418519E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    36  2.75283770E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    37  2.83149020E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    38  2.91014271E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    39  2.98879522E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    40  3.06744772E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    41  3.14610023E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    42  3.22475273E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    43  3.30340524E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    44  3.38205774E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    45  3.46071025E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    46  3.53936276E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    47  3.61801526E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    48  3.69666777E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    49  3.77532027E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    50  3.85397278E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    51  3.93262528E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    52  4.01127779E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    53  4.08993030E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    54  4.16858280E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
   1      1    55  4.24723531E-01    .0          1.00000000E 00 7.14285714E-01 1.00000000E 00 1.78571429E 00
```

Figure 21. Typical Time Line Printout.

LINE	REGION	POINT	X	U	C	P	MM	L
1	1	56	4.32568781E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	57	4.40434032E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	58	4.48311928E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	59	4.56164533E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	60	4.64029784E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	61	4.71915034E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	62	4.79780285E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	63	4.87645535E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	64	4.95510786E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	65	5.03376036E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	66	5.11241287E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	67	5.19106538E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	68	5.26971788E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	69	5.34837039E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	70	5.42702289E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	71	5.50567540E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	72	5.58432790E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	73	5.66298041E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	74	5.74163292E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	75	5.82028542E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	76	5.89893793E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	77	5.97759043E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	78	6.05624294E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	79	6.13489544E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	80	6.21354795E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	81	6.29220046E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	82	6.37085296E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	83	6.44950547E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	84	6.52815797E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	85	6.60681048E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	86	6.68546298E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	87	6.76411549E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	88	6.84276799E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	89	6.92142050E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	90	7.00007301E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	91	7.07872551E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	92	7.15737802E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	93	7.23603052E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	94	7.31468303E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	95	7.39333553E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	96	7.47198804E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	97	7.55064055E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	98	7.62929305E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	99	7.70794556E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	100	7.78659806E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	101	7.86525057E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	102	7.94390307E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	103	8.02255558E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	104	8.10120809E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	105	8.17986059E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	106	8.25851310E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	107	8.33716560E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	108	8.41581811E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	109	8.49447061E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	110	8.57312312E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00

Figure 22. Typical Time Line Printout (Continued).

LINE	REGION	POINT	X	U	C	P	MM	E
1	1	111	8.85177563E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	112	8.87342013E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	113	8.89505068E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	114	8.93577331E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	115	8.96530565E-01	.0	1.00000000E 00	7.14285714E-01	1.00000000E 00	1.78571429E 00
1	1	116	9.02227652E-01	4.50528619E-02	9.90979648E-01	6.70424152E-01	9.55712759E-01	1.75364404E 00
1	1	117	9.06130065E-01	9.27066746E-02	9.81445773E-01	6.26562590E-01	9.13626223E-01	1.72006394E 00
1	1	118	9.14415439E-01	1.43338960E-01	9.71321674E-01	5.82701028E-01	8.64528101E-01	1.68475965E 00
1	1	119	9.21110891E-01	1.97285564E-01	9.60521280E-01	5.38839465E-01	8.17629320E-01	1.64750204E 00
1	1	120	9.28290541E-01	2.55165217E-01	9.48939086E-01	4.94977903E-01	7.69523503E-01	1.60800962E 00
1	1	121	9.36037613E-01	3.17615175E-01	9.36440723E-01	4.51116341E-01	7.20182762E-01	1.56593076E 00
1	1	122	9.44459760E-01	3.85504457E-01	9.22844524E-01	4.07254779E-01	6.69421121E-01	1.52078931E 00
1	1	123	9.53369587E-01	4.59966695E-01	9.07939635E-01	3.63393217E-01	6.17110535E-01	1.47201139E 00
1	1	124	9.63942878E-01	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	125	9.71030317E-01	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	126	9.78117757E-01	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	127	9.85205196E-01	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	128	9.92292636E-01	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	129	9.99380075E-01	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	130	1.00645751E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	131	1.01355495E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	132	1.02064239E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	133	1.02772983E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	134	1.03481727E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	135	1.04190471E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	1	136	1.04899215E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
THE NEXT TWO POINTS DEFINE THE INTERFACE								
1	1	137	1.05607959E 00	5.42558170E-01	8.91403191E-01	3.19531654E-01	5.62960464E-01	1.41892795E 00
1	2	138	1.05607959E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
1	2	139	1.05867585E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
1	2	140	1.06127211E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
1	2	141	1.06386836E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
1	2	142	1.06646462E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
THE NEXT TWO POINTS DEFINE THE RIGHT TRAVELING SHOCK								
1	2	143	1.06906088E 00	5.42558170E-01	3.12611772E-01	3.19531654E-01	4.57753071E 00	1.74510929E-01
1	2	144	1.06906088E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	145	1.06990495E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	146	1.07074902E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	147	1.07159309E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	148	1.07243716E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	149	1.07328123E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	150	1.07412530E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	151	1.07496937E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	152	1.07581344E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	153	1.07665751E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02
1	2	154	1.07750158E 00	.0	1.07703141E-01	7.14285714E-03	8.62071452E-01	2.07142260E-02

(002)RIGHT SHOCK VELOCITY 6.05149392E-01

Figure 23. Typical Time Line Printout (Continued).

BEST AVAILABLE COPY

APPENDIX II - MCDU-8 LISTING

```
C    -    MAIN PROGRAM FOR BLAST WAVE.
FILF   1=BLASTA/ME2367,UNIT=DISK,BLOCKING=30,RECORD=5
FILF   2=BLASTB/ME2367,UNIT=DISK,BLOCKING=30,RECORD=5
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON/TIMUU/ DT, UU1, UU2, I,XMU
      COMMON/NOCON/ IS1, IS2, IS3, IS4, INT1, INT2, IMAX
      COMMON /INIT/ PI1,UI1,RHI1,EI1,CI1,QI1,PI2,UI2,RHI2,EI2,CI2
     1,QI2
      COMMON / REFE / TREF,T
      COMMON/DTTS/ DTT
      COMMON/NDIM/TT,TTMAX,XXZ,TMAX
      COMMON/SHKI/ EP
      COMMON/GAM/GAMM(2)
      COMMON/CON/TCON,DT2,PTUL
      COMMON/NNEWW/KSHOCK,KTELG,IDT
  901 FORMAT(I5,3E15.8)
  902 FORMAT(1H ,60(/," ",7("*"),6("MCDU8.OLD PROBLEM "),7("*")))
  903 FORMAT(1H ,60(/," ",7("*"),6("MCDU8.NEW PROBLEM "),7("*")))
  904 FORMAT(1H1,"LAST TIME LINE WILL BE PUNCHED")
  905 FORMAT(5E15.8)
  906 FORMAT(16I5)
  907 FORMAT(1H0,"NON-DIMENSIONAL INPUT DATA"/
     (1H ,"(T)TIME OF THE INITIAL TIME LINE                  ",1PE15.8/
     (1H ,"(DT)TIME INCREMENT TO THE FIRST TIME LINE         ",1PE15.8/
     (1H ,"(TMAX)MAXIMUM RUN TIME                            ",1PE15.8)
  908 FORMAT(1H0,"GAMMA FOR REGION ONE ",1PE15.8/
     (1H ,"GAMMA FOR REGION TWO ",1PE15.8)
  909 FORMAT(1H0,"(UU1)LEFT  TRAVELING SHOCK VELOCITY ",1PE15.8/
     (1H ,"(UU2)RIGHT TRAVELING SHOCK VELOCITY ",1PE15.8)
  910 FORMAT(T57,I4,T1,I5,E15.8)
  911 FORMAT(1H ,T57,I4,T1,I5,E15.8)
  913 FORMAT(1H1,"PROPERTIES OF THE INITIAL TIME LINE"/
     (1H0,"POINT NO. " ,7X,"X",14X,"U",14X,"C",14X,"P",13X,"RH",1
     14X,"E")
  914 FORMAT(1H ,3X,I4,3X,6(1PE15.8))
  927 FORMAT(1H1,6(" MCDU8.TIME LINE",I4)/
     ,1H ,"TIME OF THIS LINE                         ",1PE15.8/
     (1H ,"TIME INCREMENT TO PREVIOUS TIME LINE  ",1PE15.8/
     (1H ,"LINE REGION POINT",7X,"X",14X,"U",14X,"C",14X,"P",14X,
     1"RH",  13X,"E")
  928 FORMAT(1H1,"LINE REGION POINT",7X,"X",14X,"U",14X,"C",14X,"
     1P",     13X,"RH",14X,"E")
  929 FORMAT(1H ,"THE NEXT TWO POINTS DEFINE THE LEFT TRAVELING S
     1HOCK")
  930 FORMAT(1H ,"THE NEXT TWO POINTS DEFINE THE INTERFACE")
  931 FORMAT(1H ,"THE NEXT TWO POINTS DEFINE THE RIGHT TRAVELING
     1SHOCK")
  932 FORMAT(1H ,I4,1X,I6,1X,I5,6(1PE15.8))
  933 FORMAT(1H1,60(/,12(" END.MCDU8")))
```

```
934 FORMAT(1H1,"PUNCHED OUTPUT OF THE LAST TIME LINE")
935 FORMAT(T57,I4,T1,3E15.8)
936 FORMAT(T57,I4,T1,7I5)
938 FORMAT(" THIS CARD TO CONTAIN ISTART,IPUNCH,IDUMP,IDT,DT,TM
   1AX")
939 FORMAT(1H ,T57,I4,T2,3E15.8)
940 FORMAT(1H ,T57,I4,T2,7I5)
941 FORMAT(///1H ,10X,"TIME(T)",2X,"*",30X,"*",2X,"INTERFACE"/
   (1H ,18X,"***",28X,"***",
   (5(/,1H ,19X,"*",30X,"*")/
   (1H ,19X,"*  REGION 1",I52,"*  REGION 2",
   (4(/,1H ,19X,"*",30X,"*")/
   (1H ,19X,"*",30X,"*",29X,"*"/
   (1H ,19X,63("*")," DISTANCE FROM THE CENTER POINT (X)"/
   (1H ,80X,"*"////
   (1H0,"TITLE ABBREVIATIONS"/
   (1H0,"X -DISTANCE FROM THE CENTER POINT"/
   (1H ,"U -PARTICLE VELOCITY"/
   (1H ,"C -SOUND SPEED"/
   (1H ,"P -PRESSURE"/
   (1H ,"RH-MATERIAL DENSITY"/
   (1H ,"E -SPECIFIC ENERGY")
942 FORMAT(1H0,"(UU2)RIGHT SHOCK VELOCITY ",1PE15.8)
943 FORMAT(I2,I1,I3,I1,2E15.8)
944 FORMAT(1H ,"(UU1)LEFT SHOCK VELOCITY",1PE15.8)
946 FORMAT(////1H ,"(ISTART) ",I3/1H ,
   ("(IPUNCH) ",I3/1H ,
   ("(IDUMP ) ",I3/1H ,
   ("(IDT    ) ",I3/1H ,
   ("(DT     ) ",1PE15.8/1H ,
   ("(TMAX   ) ",1PE15.8)
    TREF = 10.
    XMU = 3.
    IREF = 1
    I = 1
    L = 1
    KDUMP=0
    READ 943,ISTART,IPUNCH,IDUMP,IDT,DT2,TMAX
    IF(ISTART.LT.1)GOTO 1
    PRINT 902
    PRINT 941
    PRINT 946,ISTART,IPUNCH,IDUMP,IDT,DT2,TMAX
    CALL READO(ISTART)
    CALL DUMP(2)
    GO TO 8054
  1 CONTINUE
    IF(ISTART.LT.0)GO TO 2
    PRINT 903
    PRINT 941
    PRINT 946,ISTART,IPUNCH,IDUMP,IDT,DT2,TMAX
```

```
      IS1=4
      IS2=4
      CALL INIDAT
      KTELG=1
      KSHOCK=1
      CALL DUMP(2)
      GO TO 8053
    2 CONTINUE
      PRINT 902
      PRINT 941
      PRINT 946,ISTART,IPUNCH,IDUMP,IDT,DT2,TMAX
      READ 905,T,DT,TCON
      TMAX=TMAX/TCON
      IF(IDT.EQ.1)DT=DT2/TCON
      READ 905,GAMM(1),GAMM(2)
      READ 905,UU1,UU2
      READ 906,IS1,IS2,IS3,IS4,IMAX
      READ 906,INT1,INT2,KTELG,KSHOCK
      PRINT 907,T,DT,TMAX
      PRINT 908,GAMM(1),GAMM(2)
      PRINT 909,UU1,UU2
      PRINT 913
      DO 3 J=1,IMAX
      READ 905,X(1,J),U(1,J),C(1,J)
      READ 905,P(1,J),RH(1,J),E(1,J)
      Q(1,J)=0.0
      PRINT 914,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J)
    3 CONTINUE
      READ 901,I,PTOL
      CALL DUMP(2)
 8054 CONTINUE
      TNEW=DT
    5 CONTINUE
      IF(KSHOCK.EQ.1)GO TO 80
      IF(UU1.GT.0.0)GO TO 80
      IF(IS1.LE.3.OR.DT*UU1+X(1,IS1).LT.X(1,3))GO TO 8
   80 CONTINUE
      CALL PTARNG
      GO TO 6
    8 CONTINUE
      EP = 0.01
      CALL REFLSK(1)
      IREF = 2
      TREF = T
      T = T + DT
      CALL SHFPT(2,IS3-2,IS3-1,IS3,IS4,IS4+1,IS4+2,UU2)
      GO TO 12
    6 CONTINUE
      T=T+DT
      TNEW = DT
```

```
      TEP = T*.215
       IF(KSHOCK.EQ.2) GO TO 6501
       IS2P2=2
      DO 6500 J=1,INT1-2
C
C
C
C PRESSURE SLOPE TEST
      TEST=(P(1,J+1)-P(1,J))/(P(1,J)*(X(1,J+1)-X(1,J)))
      IF(TEST.LT.PTOL)GO TO 6500
C
C
C
      IF((U(1,J)-C(1,J)).LE.(U(1,J+1)-C(1,J+1)))GO TO 6500
      IS1=J
       PRINT 6502,IS1
 6502 FORMAT(1H ,"A SHOCK WILL BE INSERTED AT POINT NUMBER ",I5)
      IS2=IS1+1
      CALL SWITCH(IS1+1,1,1,2)
      INT1=INT1+1
      INT2=INT2+1
      IS3=IS3+1
      IS4=IS4+1
      EP=.1*(P(1,IS1+2)-P(1,IS1))/P(1,IS1)
      CALL SHOKIN(1,2,1,IS1)
      KSHOCK=2
      GO TO 6501
 6500  CONTINUE
 6501  CONTINUE
       IF(KSHOCK.EQ.1) GO TO 6504
      CALL SHFPT(1,IS1-2,IS1-1,IS1,IS2,IS2+1,IS2+2,UU1)
      IF(P(2,IS2).LE.P(2,IS1))KSHOCK=1
      IF(KSHOCK.EQ.2) GO TO 4001
      PRINT 6503
 6503 FORMAT(1H ,"DUE TO DECAY IN PRESSURE,THE INSERTED SHOCK HAS
     1 BEEN REMOVED")
      CALL SWITCH(IS2+1,-1,1,2)
      INT1=INT1-1
      INT2=INT2-1
      IS3=IS3-1
      IS4=IS4-1
       IS1=4
       IS2=4
 4001 CONTINUE
      IS1M2 = IS1 - 2
      DO 10 K = 2, IS1M2
   10 CALL GNPT(1, K-1, K, K+1, 3)
      IF(UU1 .GE. 0.) KQ = 2
      IF(UU1 .LT. 0.) KQ = 3
      KQ = 3
```

```
      CALL GNPT(1, IS1-2, IS1-1, IS1, KQ)
   12 IF(IS2 + 1 .EQ. INT1) GO TO 21
      IF(UU1 .GE. 0.) KQ = 3
      IF(UU1 .LT. 0.) KQ = 1
      KQ = 3
      IF(KSHOCK.EQ.1) CALL GNPT(1,IS1-1,IS1,IS1+1,3)
      CALL GNPT(1, IS2, IS2+1, IS2+2, KQ)
   15 CONTINUE
      IS2P2 = IS2 + 2
 6504  CONTINUE
      INT1M1 = INT1 - 1
      IF(IS2P2 .GT. INT1M1 ) GO TO 21
      DO 20 K = IS2P2, INT1M1
   20 CALL GNPT(1, K-1, K, K+1, 3)
   21 CALL INTFPT(INT1-1, INT1, INT2, INT2+1)
      IS3M2 = IS3 - 2
      INT2P1 = INT2 + 1
      IF (INT2P1 .GT. IS3M2 ) GO TO 32
      DO 30 K = INT2P1, IS3M2
   30 CALL GNPT (2,K-1,K,K+1,3)
      IF(UU2 .GE. 0.) KQ = 2
      IF(UU2 .LT. 0.) KQ = 3
      KQ = 3
   32 CALL GNPT(2, IS3-2, IS3-1, IS3, KQ)
      IF ( UU2 .GE. 0. ) KQ = 3
      IF ( UU2 .LT. 0. ) KQ = 1
       CALL SHFPT(2,IS3-2,IS3-1,IS3,IS4,IS4+1,IS4+2,UU2)
      KQ = 3
      CALL GNPT (2,IS4,IS4+1,IS4+2,KQ)
      IS4P2 = IS4 + 2
      IMAXM1 = IMAX - 1
      DO 40 K = IS4P2, IMAXM1
   40 CALL GNPT(2, K-1, K, K+1, 3)
      IF(T .EQ. TREF) GO TO 42
      CALL CENTPT(1, 1, 2, 3)
   42 CONTINUE
      CALL ADDPTS(2)
      I = I + 1
      DO 45 J = 1, IMAX
      X(1, J) = X(2, J)
      P(1, J) = P(2, J)
      U(1, J) = U(2, J)
      RH(1, J) = RH(2, J)
      E(1, J) = E(2, J)
      C(1, J) = C(2, J)
      Q(2, J) = 0.
   45 Q(1, J) = Q(2, J)
 8053 CONTINUE
      TEMP=DT
      IF(I.EQ.1)TEMP=T
```

```
      KDUMP=KDUMP+1
      IF(KDUMP.NE.IDUMP)GO TO 120
      CALL DUMP(1)
      KDUMP=0
  120 CONTINUE
      PRINT 927,I,I,I,I,I,I,T,TEMP
      KCOUNT=0
      DO 950 J=1,IMAX
      KCOUNT=KCOUNT+1
      II=1
      IF(J.GE.INT2)II=2
      IF(J.EQ.IS1.AND.KSHOCK.EQ.2                    ) PRINT 929
      IF(J.EQ.INT1)PRINT 930
      IF(J.EQ.IS3)PRINT 931
      PRINT 932,I,II,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J)
      IF(KCOUNT.NE.55)GO TO 950
      KCOUNT=0
      PRINT 928
  950 CONTINUE
      PRINT 942,UU2
       IF(KSHOCK.EQ.1) GO TO 945
      PRINT 944,UU1
 945   CONTINUE
      IF(T.GT.TMAX)GO TO 50
      IF(I.EQ.1)GO TO 8054
      GO TO 5
   50 CONTINUE
      IF(IPUNCH.EQ.0)GO TO 961
  955 CONTINUE
       II=I
      PRINT 934
      I=1
      PUNCH 938
      PRINT 938
      I=2
      PUNCH 935,I,T,DT,TCON
      PRINT 939,I,T,DT,TCON
      I=3
      PUNCH 935,I,GAMM(1),GAMM(2)
      PRINT 939,I,GAMM(1),GAMM(2)
      I=4
      PUNCH 935,I,UU1,UU2
      PRINT 939,I,UU1,UU2
      I=5
      PUNCH 936,I,IS1,IS2,IS3,IS4,IMAX
      PRINT 940,I,IS1,IS2,IS3,IS4,IMAX
      I=6
      PUNCH 936,I,INT1,INT2,KTELG,KSHOCK
      PRINT 940,I,INT1,INT2,KTELG,KSHOCK
      DO 960 J=1,IMAX
```

```
      I=I+1
      PUNCH 935,I,X(2,J),U(2,J),C(2,J)
      PRINT 939,I,X(2,J),U(2,J),C(2,J)
      I=I+1
      PUNCH 935,I,P(2,J),RH(2,J),E(2,J)
      PRINT 939,I,P(2,J),RH(2,J),E(2,J)
  960 CONTINUE
      I=I+1
      PUNCH 910,I,II,PTUL
      PRINT 911,I,II,PTUL
  961 CONTINUE
      PRINT 933
   60 CONTINUE
       STOP
      END



      SUBROUTINE DUMP(N)
      COMMON/GAIN/Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,10
     100)     ,E(2,1000),P(2,1000)
      COMMON/REFL/TREF,T
      COMMON/TIMUU/DT,UU1,UU2,I,XMU
      COMMON/NOCON/IS1,IS2,IS3,IS4,INT1,INT2,IMAX
      COMMON/NDIM/TT,TTMAX,XXZ,TMAX
      COMMON/GAM/GAMM(2)
      COMMON/CON/TCON,DT2
      COMMON/NNEWW/KSHOCK,KTELG,IDT
      REWIND N
      WRITE(N)T,DT,TCON
      WRITE(N)GAMM(1),GAMM(2)
      WRITE(N)UU1,UU2
      WRITE(N)IS1,IS2,IS3,IS4,IMAX
      WRITE(N)INT1,INT2,KTELG,KSHOCK
      DO 9500 J=1,IMAX
      WRITE(N)X(1,J),U(1,J),C(1,J)
      WRITE(N)P(1,J),RH(1,J),E(1,J)
 9500 CONTINUE
      WRITE(N)I,PTUL
      REWIND N
      RETURN
      END



      SUBROUTINE READU(N)
      COMMON/GAIN/Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,10
     100)     ,E(2,1000),P(2,1000)
      COMMON/REFL/TREF,T
      COMMON/TIMUU/DT,UU1,UU2,I,XMU
```

```
      COMMON/NOCON/IS1,IS2,IS3,IS4,INT1,INT2,IMAX
      COMMON/NDIM/TT,TTMAX,XXZ,TMAX
      COMMON/GAM/GAMM(2)
      COMMON/CON/TCON,DT2,PTOL
      COMMON/NNEWW/KSHOCK,KTELG,IDT
  907 FORMAT(1H0,"NON-DIMENSIONAL INPUT DATA"/
     (1H ,"(T)TIME OF THE INITIAL TIME LINE                ",1PE15.8/
     (1H ,"(DT)TIME INCREMENT TO THE FIRST TIME LINE       ",1PE15.8/
     (1H ,"(TMAX)MAXIMUM RUN TIME                          ",1PE15.8)
  908 FORMAT(1H0,"GAMMA FOR REGION ONE ",1PE15.8/
     (1H ,"GAMMA FOR REGION TWO ",1PE15.8)
  909 FORMAT(1H0,"(UU1)LEFT  TRAVELING SHOCK VELOCITY ",1PE15.8/
     (1H ,"(UU2)RIGHT TRAVELING SHOCK VELOCITY ",1PE15.8)
  913 FORMAT(1H1,"PROPERTIES OF THE INITIAL TIME LINE"/
     (1H0,"POINT NO. " ,7X,"X",14X,"U",14X,"C",14X,"P",13X,"RH",1
     14X,"E")
  914 FORMAT(1H ,3X,I4,3X,6(1PE15.8))
  925 FORMAT(1H0,"(PTOL)SHOCK PRESSURE JUMP TOLERANCE ",1PE15.8)
      REWIND N
      READ (N)T,DT,TCON
      TMAX=TMAX/TCON
      IF(IDT.EQ.1)DT=DT2/TCON
      READ (N)GAMM(1),GAMM(2)
      READ (N)UU1,UU2
      READ (N)IS1,IS2,IS3,IS4,IMAX
      READ (N)INT1,INT2,KTELG,KSHOCK
      PRINT 907,T,DT,TMAX
      PRINT 908,GAMM(1),GAMM(2)
      PRINT 909,UU1,UU2
      PRINT 913
      DO 9500 J=1,IMAX
      READ (N)X(1,J),U(1,J),C(1,J)
      READ (N)P(1,J),RH(1,J),E(1,J)
      Q(1,J)=0.0
      PRINT 914,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J)
 9500 CONTINUE
      READ (N)I,PTOL
      PRINT 925,PTOL
      REWIND N
      RETURN
      END


      SUBROUTINE ADDPTS(L)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),    E(2,1000),P(2,1000)
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/NOCON/ IS1, IS2, IS3, IS4, INT1, INT2, IMAX
      COMMON/INIT/PI1,UI1,RHI1,EI1,CI1,QI1,PI2,UI2,RHI2,EI2,CI2,Q
```

```
     112
      COMMON/SINPT/ PI, UI, RHI, UF, XZ, PF, IR1, IR2
      COMMON / DTTS / DTT
C 800 FORMAT("  ADD POINTS TO THE REGION RIGHT TO MAIN SHOCK")
  802 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X)," ADDED PTS")
       ISPEC=IMAX-1
      IS4P1 = IS4 + 1
      IMAX = IS4 + 10
       IF(ABS(DT) .LT. 0.0001 ) GO TO 5
C     IF ( I .EQ. 1) DX = DT/CI2
       IF(I.EQ.1) DX=DT/C(2,ISPEC)*50.
      IF(I.GE.2)DX=DTT/C(2,ISPEC)*50.
C     IF( I .GE. 2) DX = DTT/CI2
    5  DX=2.0
      DO 10 J = IS4P1, IMAX
      X(2, J) = X(2, J-1) + DX
       P(2,J)=P(2,ISPEC)
       U(2,J)=U(2,ISPEC)
       E(2,J)=E(2,ISPEC)
       C(2,J)=C(2,ISPEC)
       RH(2,J)=RH(2,ISPEC)
       Q(2,J)=Q(2,ISPEC)
   10 CONTINUE
      RETURN
      END


      FUNCTION ADT (M,N)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON /TIMUB/ DT,UU1,UU2,I, XMU
  101 FORMAT (1H ,"DT IS NEGATIVE",2(5X,I3))
      RA = 0.99
      DX = X(1,M) - X(1,N)
      AT = DX/(C(1,M) + C(1,N)) * 2.0*RA
      IF ( AT .LE. 0.) PRINT 101,M,N
      ADT = AT
      RETURN
      END


      SUBROUTINE ASIGPT
C      TO ASIGN PTS FOR THE INITIAL LINE, FOR MSR = 1 CASE
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON/RAWAV/ XR(30),UR(30),CR(30),RHR(30),ER(30),PR(30),RH
     1P(30)
      COMMON /C1AND2/ PF1,UF1,RHF1,EF1,CF1,QF1,PF2,UF2,RHF2,EF2,C
```

```
     1F2,QF2,   XF,XS
      COMMON / INIT/ PI1,UI1,RHI1,EI1,CI1,QI1,PI2,UI2,RHI2,EI2,CI
     12,QI2
      COMMON/NOCON/ IS1, IS2, IS3, IS4, INT1, INT2, IMAX
      COMMON/SINPT/ PI, UI, RHI, UF, XZ, PF, IR1, IR2
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/SHKI/ EP
C       IR1 = FIRST PT IN RAWAVE;  IR2 = LAST PT IN RAWAVE
  800 FORMAT (1H ,1X,"I","    J",10X,"X",17X,"U",16X,"C",16X,"P",1
     16X,"RH"    ,16X,"E",10X,"L")
  801 FORMAT (1H ,I2,1X,I4,3X,6(E15.8,2X),I2)
  802 FORMAT (1H ,9X,"Q   = ",E15.8)
  810 FORMAT("  INITIAL DATA BEING READ IN AND ARRANGED IN ORDER"
     1,         /, "  SECOND SHOCK ALSO BEEN INSERTED AT IS1 AND I
     2S2")
  811 FORMAT("  IS1 =", I4, "  IS2 =", I4, "  IS3 =", I4, "  IS4
     1=", I4,  /, "  INT1 =", I4, "  INT2 =", I4, "  IMAX =", I4)
  812 FORMAT (1H0," END OF INITIAL DATA ARRANGEMENT FOR BLAST WAV
     1E, READY FOR STARTING CALCULATION")
C       CALCULATE AVERAGE DT IN THE RAREFACTION FAN
      DT1 = 0.
      ISHOK = 1
      IR1P = IR1
      IR2P = IR2
      IR12 = IR2 - IR1
      DO 5 N=IR1,IR2
    5 X(1, N) = XR(N)
      DO 10 M=2,IR2
      N=M-1
      DT1= ADT ( M,N) + DT1
   10 CONTINUE
      XIR2 = IR2 - IR1
      AVDT = DT1/XIR2
C       CALCULATE DX AND VARIABLES BETWEEN CENTER LINE AND
C         RAREFACTION WA
      N1 = X(1, IR1)/(CI1*AVDT)
C     PRINT 816,IR1,IR2,DTI,AVDT,CI1,X(1,IR1),N1
C 816 FORMAT(////,"IR1=",I3,"IR2=",I3,"DTI=",E13.6,"AVDT=",E13.6,
C        "CI1=",
C    $ E13.6," X(1,IR1)=",E13.6,"N1=",I4)
      IR1 = N1+1
      IR2 = IR1 + IR12
      DO 15 N=IR1,IR2
      X(1, N) = XR(  IR1P + N - IR1)
      P(1, N) = PR(  IR1P + N - IR1)
      U(1, N) = UR(  IR1P+N-IR1)
C     PRINT 817,N,X(1,N),P(1,N),U(1,N)
C
C 817 FORMAT(1H ,"N=",I4,"X(1,N)=",E13.6,"P(1,N)=",E13.6,"U(1,
C        N)=",E13.6
```

```
C      $)
       RH(1, N) = RHR(  IR1P+N-IR1)
       E(1, N) = ER(  IR1P+N-IR1)
       C(1, N) = CR(  IR1P+N-IR1)
    15 Q(1, N) = 0.
       XN1 = N1
       DX = X(1,IR1)/XN1
       X(1,1)=0.
       IR1M1 = IR1 - 1
       DO 20 N=2,IR1M1
       X(1,N)=X(1,N-1) + DX
    20 CONTINUE
       DO 25 N=1,IR1M1
       P(1,N) = PI1
       U(1,N) = UI1
       RH(1,N) = RHI1
       E(1,N) = EI1
       C(1,N) = CI1
       Q(1,N) = QI1
    25 CONTINUE
C   -   INSERT SHOCK AT PT IR2.
C      EP = 0.
C      CALL SHOKIN ( 1, 2, 3, IR2 )
C      GO TO 27
C      EP = (P(1,IR2-1) - P(1,IR2))/P(1,IR2) * 0.1
C      CALL SHOKIN(1, 1, 1, IR2)
    27 CONTINUE
C       COMPUTE DX AND VARIABLES BETWEEN RAWAVE TAIL TO INTFPT
       N2 = (XF - XR(IR2P))/(CF1*AVDT)
       XN2 = N2
       IF (N2 .LE. 1 ) GO TO 32
       DX = (XF - XR(IR2P))/XN2
C      INT1 = IS2 + N2
        INT1=IR2+N2
       INT2 = INT1 + 1
       INT1M1 = INT1 - 1
C      IS2P1 = IS2 + 1
       IR2P1 = IR2 + 1
       KING = 1
C      DO 30 N = IS2P1, INT1M1
       N = IR2P1
    28 IF(ISHOK .NE. KING) GO TO 29
       NM1 = N - 1
C   -   INSERT SHOCK AT POINT NM1.
C      EP = 0.
C      CALL SHOKIN(1, 2, 3, NM1)
          EP = 0.001
    29 CONTINUE
       X(1,N) = X(1,N-1) + DX
       P(1, N) = PF1
```

```
      U(1, N) = UF1
      RH(1, N) = RHF1
      E(1, N) = EF1
      C(1, N) = CF1
      Q(1, N) = QF1
      KING = KING + 1
      IF(N .EQ. INT1M1) GO TU 30
      N = N + 1
      GO TO 28
   30 CONTINUE
      GO TO 35
   32 INT1 = IS2 + 1
      INT2 = INT1 + 1
C      COMPUTE DX AND VARIABLES BETWEEN INTFPT AND SHOCKFPT
   35 CONTINUE
      X(1,INT1) = XF
      P(1,INT1) = PF1
      U(1,INT1) = UF1
      RH(1,INT1) = RHF1
      E(1,INT1) = EF1
      C(1,INT1) = CF1
      Q(1,INT1) = QF1
      X(1,INT2) = XF
      P(1,INT2)= PF2
      U(1,INT2) = UF2
      RH(1,INT2) = RHF2
      E(1,INT2 ) = EF2
      C(1,INT2) = CF2
      Q(1,INT2) = QF2
      N3 = (XS -XF) /(CF2*AVDT)
      XN3 = N3
      IF (N3.LE. 1) GO TO 42
      DX = (XS - XF)/XN3
      IS3 = INT2 + N3
      IS4 = IS3 + 1
      INT2P1 = INT2 + 1
      IS3M1 = IS3 - 1
      DO 40 N=INT2P1,IS3M1
      X(1,N ) = X(1,N-1) + DX
      P(1,N) = PF2
      U(1,N) = UF2
      RH(1,N) = RHF2
      E(1,N) = EF2
      C(1,N) = CF2
      Q(1,N) = QF2
   40 CONTINUE
      GO TO 45
   42 IS3 = INT1 + 1
      IS4 = IS3 + 1
   45 CONTINUE
```

```
      X(1,IS3) = XS
      P(1,IS3) = PF2
      U(1,IS3) = UF2
      RH(1,IS3) = RHF2
      E(1,IS3)= EF2
      C(1,IS3) = CF2
      Q(1,IS3) = QF2
      X(1,IS4) = XS
      P(1,IS4) = PI2
      U(1,IS4) = UI2
      RH (1,IS4) = RHI2
      E(1,IS4) =EI2
      C(1,IS4) = CI2
      Q(1,IS4) = QI2
C      ADD POINTS IN REGION 2    (10 PTS)
      IS4P1 = IS4 + 1
      IMAX = IS4 + 10
      DX = CI2 * AVDT
      DO 50 N = IS4P1,IMAX
      X(1,N) = X(1,N-1) + DX
      P(1,N) = PI2
      U(1,N) = UI2
      RH(1,N) = RHI2
      E(1,N) = EI2
      C(1,N) = CI2
      Q(1,N) = QI2
   50 CONTINUE
      DT = AVDT
      L = 1
      I = 1
      RETURN
      END



      SUBROUTINE CALCDT
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON /NUCON/ IS1,IS2,IS3,IS4,INT1,INT2,IMAX
      COMMON/TIMUU/ DT, UU1, UU2, I,XMU
C 100 FORMAT (1H ,10X,"DT  = ", E15.8)
      SMDT = 100.
      IF(INT1 .EQ. 0) GO TO 26
      DO 10 K = 2,IS1
      XADT = ADT(K,K-1)
   10 IF (XADT .LT. SMDT) SMDT = XADT
      IS2P1 = IS2 + 1
      DO 15 K = IS2P1,INT1
      XADT = ADT (K,K-1)
   15 IF (XADT .LT. SMDT) SMDT = XADT
```

```
      INT2P1 = INT2 + 1
      DO 20 K = INT2P1,IS3
       XADT = ADT (K,K-1)
   20 IF (XADT .LT. SMDT) SMDT = XADT
      IS4P1 = IS4 + 1
      DO 25 K = IS4P1,IMAX
      XADT = ADT(K,K-1)
   25 IF (XADT .LT. SMDT) SMDT = XADT
      GO TO 28
   26 DO 27 K=2,IMAX
      XADT= ADT(K,K-1)
   27 IF( XADT .LT. SMDT) SMDT=XADT
   28 CONTINUE
      DT = SMDT
C     PRINT 100,DT
      RETURN
      END



      SUBROUTINE CENTPT(L, I1, I2, I3)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
  100 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X), I2," CENTER")
  101 FORMAT(11X, " NOT CONVERGE")
C 102 FORMAT(1H ," CHECK PRINT    K=    ",I3)
C2000 FORMAT(6X, " XA", 10X, "UA", 10X, "PA", 10X, "RHA", 9X,
C      "EA",/,
C    1  1H , 5E11.4)
      XINP(V1, DV, DX, DY) = V1 + DV*DY/DX
      LIM = 30
      TOL = 0.0005
      TOL1 = 1.E-10
      K = 1
C   -  DEFINE VARIABLES.
      U3 = 0.
      X3 = 0.
      U(1, I1) = 0.
      X (2,I1) = 0.0
      U4 = U(2, I2)
      U5 = U(2, I3)
      DX2 = X(2, I2) - X(2, I1)
      DX3 = X(2, I3) - X(2, I1)
      UX3 = (U4*DX3**2 - U5*DX2**2)/(DX2*DX3*(DX3 - DX2))
      P1 = P(1, I1)
      RH1 = RH(1, I1)
      C1 = C(1, I1)
      U1 = 0.
      E1 = E(1, I1)
```

```
      Q1 = Q(1, I1)
      P2 = P(1, I2)
      RH2 = RH(1, I2)
      C2 = C(1, I2)
      U2 = U(1, I2)
      Q3 = Q(1, I2)
      DX1 = X(1, I2) - X(1, I1)
      DU1 = U(1, I2) - U(1, I1)
      DRH1 = RH(1, I2) - RH(1, I1)
      DC1 = C(1, I2) - C(1, I1)
      DE1 = E(1, I2) - E(1, I1)
      DP1 = P(1, I2) - P(1, I1)
      DQ1 = Q(1, I2) - Q(1, I1)
C   -  ASSUME PROPERTIES AT POINTS A
      PA = (P1 + P2)/2.
      RHA = (RH1 + RH2)/2.
      CA = (C1 + C2)/2.
      UA = U2
      E3 = E1
      C3 = C2
      RH3 = RH1
      P3 = P1
C   -  BEGINNING OF ITERATION.
   10 CONTINUE
      UMC3A = (U3 + UA - C3 - CA)/2.
      XA = -UMC3A*DT
      UA = XINP(U1, DU1, DX1, XA)
      CA = XINP(C1, DC1, DX1, XA)
      EA = XINP(E1, DE1, DX1, XA)
      PA = XINP(P1, DP1, DX1, XA)
      RHA = EQSTRQ(L, EA, PA)
      QA = XINP(Q1, DQ1, DX1, XA)
      RHC3A = (RH3*C3 + RHA*CA)/2.
      UCX3A = (UA*CA/XA + UX3*C3)/2.
      PE3 = EQSTPE(L, E3, RH3)
      PEA = EQSTPE(L, EA, PA)
      PRCQ3A = (PE3*Q3/(RH3*C3) + PEA*QA/(RHA*CA))/2.
      PR13 = (P1/(RH1**2) + P3/(RH3**2))/2.
      Q13 = (Q1 + Q3)/2.
      P3P= PA-UA*RHC3A+(-(XMU-1.) *UCX3A+PRCQ3A) *RHC3A*DT
      E3P= E1+ PR13*(RH3-RH1)+ Q13*DT
      RH3P= EQSTRQ( L,E3P,P3P)
      C3P= EQSTCQ (L,E3P,RH3P,P3P)
      IF (ABS((P3P-P3)/P3)- TOL)21,21,40
   21 IF (ABS((E3P-E3)/E3)- TOL)22,22,40
   22 IF (ABS((RH3P-RH3)/RH3) - TOL)23,23,40
   23 IF (ABS((C3P-C3)/C3)- TOL) 50,50,40
   40 CONTINUE
      K= K+1
      P3= (P3+P3P)/2.
```

```
      E3= (E3+E3P)/2.
      RH3= (RH3+RH3P)/2.
      C3= (C3+C3P)/2.
C     PRINT 2000, XA, UA, PA, RHA, EA
C     PRINT 100,I,I1,X(1,I1),U3,C3,P3,RH3,E3,L
C     PRINT 102,K
      IF(K.LT.LIM) GO TO 45
      PRINT 101
      GO TO 50
   45 CONTINUE
      GO TO 10
   50 CONTINUE
      X(2, I1) = 0.
      U(2, I1) = 0.
      C(2, I1) = C3P
      RH(2, I1) = RH3P
      E(2, I1) = E3P
      P(2, I1) = P3P
      RETURN
      END


      FUNCTION EQSTCQ(L, E, RH, P)
      COMMON/GAM/GAMM(2)
  101 FORMAT(1H0," RH=",E15.8," P=",E15.8)
      IF(RH .LT.0.) PRINT 101,RH,P
      IF(P .LT. 0.) PRINT 101,RH,P
      GO TO (10,20),L
   10 CONTINUE
C      FUNCTION TO CALCULATE SPEED OF SOUND
      GAMMA=GAMM(1)
      C2=GAMMA*P/RH
       IF ( C2 .LE. 0 ) C2= -C2
      EQSTCQ = SQRT(C2)
      RETURN
   20 CONTINUE
      GAMMA=GAMM(2)
      C2=GAMMA*P/RH
       IF ( C2 .LE. 0 ) C2= -C2
      EQSTCQ=SQRT(C2)
      RETURN
      END


      FUNCTION EQSTEQ(L, RH, P)
      COMMON/GAM/GAMM(2)
      GO TO (10,20),L
   10 CONTINUE
```

```
      GAMMA=GAMM(1)
      EQSTEQ = P/((GAMMA - 1.)*RH)
      RETURN
   20 CONTINUE
      GAMMA=GAMM(2)
      EQSTEQ=P/((GAMMA-1.0)*RH)
      RETURN
      END


      FUNCTION EQSTRQ (L,E,P)
      COMMON/GAM/GAMM(2)
      GO TO (10,20),L
   10 CONTINUE
C       FUNCTION TO CALCULATE DENSITY FROM EQ. STATE
      GAMMA=GAMM(1)
      EQSTRQ=P/((GAMMA-1.0)*E)
      RETURN
   20 CONTINUE
      GAMMA=GAMM(2)
      EQSTRQ=P/((GAMMA-1.0)*E)
      RETURN
      END


      FUNCTION EQSTPE(L,E,RH)
      COMMON/GAM/GAMM(2)
      GO TO (10,20),L
C       FUNCTION TO CAL. DERIVATIVE OF P W.R. TO E FOR RH CONSTANT
   10 CONTINUE
      GAMMA=GAMM(1)
      EQSTPE=(GAMMA-1.0)*RH
      RETURN
   20 CONTINUE
      GAMMA=GAMM(2)
      EQSTPE=(GAMMA-1.0)*RH
      RETURN
      END


      FUNCTION EQSTPQ(L, E, RH)
      COMMON/GAM/GAMM(2)
C       FUNCTION TO CALCULATE PRESSURE FROM EQ. STATE
      GO TO (10,20),L
   10 CONTINUE
      GAMMA=GAMM(1)
      EQSTPQ = (GAMMA - 1.0)*RH*E
```

```
      IF (EQSTPQ .LT. 0. ) EQSTPQ = 0.
      RETURN
   20 CONTINUE
      GAMMA=GAMM(2)
      EQSTPQ =(GAMMA - 1.0)*RH*E
      IF(EQSTPQ .LT. 0.)  EQSTPQ= 0.
      RETURN
      END



      FUNCTION EQSTPR (L,E,RH)
      COMMON/GAM/GAMM(2)
      GO TO (10,20),L
   10 CONTINUE
C      FUNCTION TO CAL. DER. OF P W.R. TO RH FOR E CONSTANT
      GAMMA=GAMM(1)
      EQSTPR =(GAMMA-1.0)*E
      RETURN
   20 CONTINUE
      GAMMA=GAMM(2)
      EQSTPR=(GAMMA-1.0)*E
      RETURN
      END



       SUBROUTINE CALENG(XMAX)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),    E(2,1000),P(2,1000)
      DIMENSION SUM1(1000),SUM2(1000),SUM3(1000)
       DIMENSION FF(2,500),GG(2,500),HH(2,500)
  101 FORMAT(1H ," KEN ENERGY=",E15.8,"   INT ENERGY=", E15.8,
     1 "  TOTOL ENERGY=", E15.8, "IMAX=", I5)
  102  FORMAT(1H ,3E15.8)
 1000 FORMAT(4E15.8)
      U0=0.7781E 04
       U0=0.807263E 04
      I= 2
      I= 1
C   *****    D=7781 M/SEC.    ****   RHCJ=2.237 GRAM/(CM)**3
C        *********
      D=7.781E 03
C   FOR D IN M/SEC,TOTOL ENERGY IN CAL./GRAM,THEN CKEN=0.239E-03,
C        CINT=1.0
      CKEN=0.239E-03
      CINT=1.E0
      U02= U0 **2
       XX2=X(I,1)**2
       UU2=U(I,1)**2
```

```
      FF(I,1)= E(I,1)*RH(I,1)*XX2
      GG(I,1)= 0.5E0*UU2*RH(I,1)*UU2*XX2
      HH(I,1)= RH(I,1)* XX2
      SUM1(1)=0.E0
      SUM2(1)=0.E0
      SUM3(1)=0.E0
      J=2
   10 JM1=J-1
   11 DX=X(I,J)-X(I,JM1)
      IF(DX  .GT. 0.1E-04) GO TO 12
      J=J+1
      GO TO 11
   12 XX2= X(I,J)**2
      UU2=U(I,J)**2
      FF(I,J)= E(I,J) * RH(I,J) * XX2
      GG(I,J)= 0.5E0*UU2*RH(I,J)*UU2*XX2
      HH(I,J)= RH(I,J) * XX2
      SUM1(J)= SUM1(JM1) + 0.5E0*(FF(I,JM1) + FF(I,J)) *DX
      SUM2(J)= SUM2(JM1) + 0.5E0*(GG(I,JM1) + GG(I,J)) *DX
      SUM3(J)= SUM3(JM1) + 0.5E0*(HH(I,JM1) + HH(I,J)) *DX
      PRINT 102,SUM1(J),SUM2(J),SUM3(J)
      IF(  ABS(X(I,J)-XMAX)  .LT. 0.1E-04) GO TO 20
      J=J+1
       GO TO 10
   20 SUMINT=SUM1(J)/SUM3(J) *CINT
      SUMKEN=SUM2(J)/SUM3(J) *CKEN
       IMAX=J
      ENERGY= SUMINT+ SUMKEN
      RETURN
      END
      SUBROUTINE GNPSHA ( L, I1, I2, I3, X4 )
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),    E(2,1000),P(2,1000)
      COMMON /SHK4A / U4A,C4A,RH4A,E4A,P4A,X4A,Q4A
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      QINP(V2, DV1, DV2, Y) = V2 + (DV2*DX1**2 + DV1*DX2**2)*Y/(D
     1X1*DX2*    (DX1 + DX2)) + (-DV1*DX2 + DV2*DX1)*Y**2/(DX1*DX2
     2*(DX1 + DX2))
 1000 FORMAT (1H ,"POINT 4A DOES NOT CONVERGE")
C1001 FORMAT (1H ,5(E11.4))
C1002 FORMAT (1H0,"   P4P    U4P    E4P    RH4P    C4P")
C1003 FORMAT(1H ,"XA = ",E15.8," XB = ",E15.8,"XC = ",E15.8,"UA
C        = ",
C     $    E15.8)
C2000 FORMAT("  A ", 6(E15.8, 2X))
C2001 FORMAT("  B ", 6(E15.8, 2X))
C2002 FORMAT("  C ", 6(E15.8, 2X))
C     PRINT 1002
      TOL = 0.0005
      TOL1=1.E-20
```

```
      K = 1
C   -   INITIAL ASSUMPTIONS FOR PT 4.
      U4 = U(1, I2)
      C4 = C(1, I2)
      RH4 = RH(1, I2)
      E4 = E(1, I2)
      P4 = P(1, I2)
      UA = U(1, I3)
      CA = C(1, I3)
      UB = U(1, I1)
      CB = C(1, I1)
      UC = U(1, I2)
      CC = C(1, I2)
C   -   DEFINE VARIABLES.
      RH2 = RH(1, I2)
      C2 = C(1, I2)
      U2 = U(1, I2)
      P2 = P(1, I2)
      E2 = E(1, I2)
      Q2 = Q(1, I2)
      DX1 = X(1, I2) - X(1, I1)
      DU1 = U(1, I2) - U(1, I1)
      DC1 = C(1, I2) - C(1, I1)
      DP1=P(1,I2)-P(1,I1)
      DRH1 = RH(1, I2) - RH(1, I1)
      DE1 = E(1, I2) - E(1, I1)
      DQ1 = Q(1, I2) - Q(1, I1)
      DX2 = X(1, I3) - X(1, I2)
      DU2 = U(1, I3) - U(1, I2)
      DC2 = C(1, I3) - C(1, I2)
      DP2=P(1,I3)-P(1,I2)
      DRH2 = RH(1, I3) - RH(1, I2)
      DE2 = E(1, I3) - E(1, I2)
      DQ2 = Q(1, I3) - Q(1, I2)
C   -   ESTIMATE POSITIONS FOR PTS A, B, AND C.
   10 CONTINUE
      XA = X4 - (U4 - C4 + UA - CA)/2.*DT
      XB = X4 - (U4 + C4 + UB + CB)/2.*DT
      XC = X4 - (U4 + UC)/2.*DT
      DXA = XA - X(1, I2)
      DXB = XB - X(1, I2)
      DXC= XC - X ( 1 ,I2 )
C     PRINT 1003 ,XA,XB,XC,UA
      PA=QINP(P2,DP1,DP2,DXA)
      EA = QINP(E2, DE1, DE2, DXA)
      RHA=EQSTRQ(L,EA,PA)
C     PRINT 2000, PA, EA, RHA, CA, UA, XA
      IF (RHA .LT. 0. ) RHA=-0.5D0*RHA
      CA = EQSTCQ(L,EA,RHA,PA)
      UA = QINP(U2, DU1, DU2, DXA)
```

```
      PEA = EQSTPE(L, EA, RHA)
      QA = QINP(Q2, DQ1, DQ2, DXA)
      PB=QINP( P2,DP1,DP2,DXB)
      EB = QINP(E2, DE1, DE2, DXB)
      RHB=EQSTRQ(L,EB,PB)
C     PRINT 2001, PB, EB, RHB, CB, UB, XB
      IF (RHB .LT. 0. ) RHB=-0.5D0*RHB
      CB = EQSTCQ (L,EB,RHB,PB)
      UB = QINP(U2, DU1, DU2, DXB)
      PEB = EQSTPE(L, EB, RHB)
      QB = QINP(Q2, DQ1, DQ2, DXB)
      EC = QINP(E2, DE1, DE2, DXC)
      PC = QINP ( P2,DP1,DP2,DXC)
      RHC = EQSTRQ (L,EC,PC)
      QC = QINP(Q2, DQ1, DQ2, DXC)
      PE4 = EQSTPE ( L, E4 , RH4 )
      Q4 = QC
C   -   CALCULATE COEFF. IN THE CHARAC. EQ..
      RHC4A = (RH4*C4 + RHA*CA)/2.
      RHC4B = (RH4*C4 + RHB*CB)/2.
      UCX4B = (U4*C4/X4 + UB*CB/XB)/2.
      UCX4A = (U4*C4/X4 + UA*CA/XA)/2.
      P RCQ4B = (PE4*Q4/(RH4*C4) + PEB*QB/(RHB*CB))/2.
      P RCQ4A = (PE4*Q4/(RH4*C4) + PEA*QA/(RHA*CA))/2.
      PEQ4A = (PE4*Q4 + PEA*QA)/2.
      PEQ4B = (PE4*Q4 + PEB*QB)/2.
      PRH4C = (P4/RH4**2 + PC/RHC**2)/2.
      QC4 = QC
      P4P = (PB/RHC4B + PA/RHC4A + UB - UA + (-(XMU - 1.)*(UCX4B
     1+         UCX4A) + P RCQ4B + P RCQ4A)*DT)/(1./RHC4A + 1./RH
     2C4B)
      U4P = (PB - PA + RHC4B*UB + RHC4A*UA + (-(XMU - 1.)* UCX4B*
     1RHC4B +    PEQ4B + (XMU - 1.)*UCX4A*RHC4A - PEQ4A)*DT)/(RHC4
     2B + RHC4A)
      E4P = EC + PRH4C*(RH4 - RH2) + QC4*DT
      RH4P = EQSTRQ(L, E4P, P4P)
C     PRINT 1001,P4P,U4P,E4P,RH4P,C4P
      IF(RH4P .LT.0. ) RH4P=-0.5D0*RH4P
      C4P = EQSTCQ(L, E4P, RH4P, P4P)
C     PRINT 1001,P4P,U4P,E4P,RH4P,C4P
      IF ( ABS (P4P).LE. TOL1)GO TO 15
      IF ( ABS ((P4P-P4)/P4P).GT. TOL) GO TO 20
   15 IF ( ABS (U4P) .LE. TOL1)GO TO 18
      IF ( ABS ((U4P - U4)/U4P).GT. TOL ) GO TO 20
   18 IF (ABS(E4P).LE. TOL1) GO TO 20
      IF ( ABS ((E4P-E4)/E4P) .GT. TOL) GO TO 20
      GO TO 30
   20 IF(K .GE. 20) GO TO 35
      K = K + 1
       U4=(U4P+U4)*0.5
```

```
      C4=(C4P+C4)*0.5
      P4=(P4P+P4)*0.5
      RH4=(RH4P+RH4)*0.5
      E4=(E4P+E4)*0.5
      GO TO 10
   15 PRINT 1000
   30 CONTINUE
      X4A = X4
      U4A = U4P
      C4A = C4P
      RH4A = RH4P
      E4A = E4P
      P4A = P4P
      Q4A = Q4
C     PRINT 2000, PA, EA, RHA, CA, UA, XA
C     PRINT 2001, PB, EB, RHB, CB, UB, XB
C     PRINT 2002, PC, EC, RHC, CC, UC, XC
C     PRINT 1003 ,XA,XB,XC,UA
      RETURN
      END


      SUBROUTINE GNPSHB(L, I1, I2, I3, X4, MS)
      COMMON/GAIN/ Q(2,1000)*X(2,1000),U(2,1000),C(2,1000),RH(2,1
     11000),    E(2,1000),P(2*1000)
      COMMON / SHK4B / U4B,C4B,RH4B,E4B,P4B,X4B,Q4B     ,UUU
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/GNPSBB/ XB
C 100 FORMAT(6X, "XB", 10X, "UB",10X, "PB",10X,"RHB",9X,"EB",/,
C        1H ,
C     1  5E11.4)
C 101 FORMAT(6X, "P4B =", E11.4, 5X, "U4B =", E11.4)
      XINP(V1, DV, DX, DY) = V1 + DV*DY/DX
C   -   MS=1,RIGHT RUNNING SHOCK;  MS = 2,LEFT RUNNING SHOCK.
      SIGN = 1.
      IF(MS .EQ. 1) SIGN = -1.
      X2 = X(1, I2)
      Q1 = Q(1, I2)
      P1 = P(1, I2)
      U1 = U(1, I2)
      E1 = E(1, I2)
      DXB = XB - X(1, I2)
      IF(XB .LT. X2) GO TO 10
      DX = X(1, I3) - X(1, I2)
      DQ1 = Q(1, I3) - Q(1, I2)
      DP1 = P(1, I3) - P(1, I2)
      DE1 = E(1, I3) - E(1, I2)
      DU1 = U(1, I3) - U(1, I2)
      GO TO 20
```

```
   10 CONTINUE
      DX = X(1, I1) - X(1, I2)
      DP1 = P(1, I1) - P(1, I2)
      DE1 = E(1, I1) - E(1, I2)
      DU1 = U(1, I1) - U(1, I2)
      DQ1 = Q(1, I1) - Q(1, I2)
   20 CONTINUE
      PB = XINP(P1, DP1, DX, DXB)
      EB = XINP(E1, DE1, DX, DXB)
      RHB = EQSTRO(L, EB, PB)
      CB = EQSTCQ(L, EB, RHB, PB)
      UB = XINP(U1, DU1, DX, DXB)
      QB = XINP(Q1, DQ1, DX, DXB)
      PEB = EQSTPE(L, EB, RHB)
      RHC = RHB*CB
      UCX = UB*CB/XB
      PRCQ = PEB*QB/RHC
      Q4B = 0.
      P4B = PB +(SIGN*(U4B - UB) + (-(XMU - 1.)*UCX + PRCQ)*DT) *
     1 RHC
       IF (P4B .LE. 0. ) P4B=-P4B
C     PRINT 100, XB, UB, PB, RHB, EB
C     PRINT 101, P4B, U4B
      RETURN
      END



      SUBROUTINE GNPT(L, I1, I2, I3, KQ)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
C   -   CALCULATE PROPERTIES AT GENERAL POINTS.
      COMMON/TIMUU/ DT, UU1, UU2, I,XMU
      REAL LINP
  100 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X), I2, 2X, 2HM1)
  101 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X), I2, 2X, 2HM2)
  103 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X), I2)
    1 FORMAT(1H , 10X, "NOT CONVERGE")
  300 FORMAT(1H , "PT M1 CALC.  PT B LIES ON LEFT  RUNNING SHOCK.
     1")
  301 FORMAT(1H , "PT M2 CALC.  PT A LIES ON RIGHT RUNNING SHOCK"
     1)
C 400 FORMAT(1H ,"XB   = ",E15.8,"  DT1   = ",E15.8)
  401 FORMAT(1H ,"1ST CHARACTERISTIC DOES NOT INTERSECT THE SHOCK
     1")
  402 FORMAT(1H ,"2ND CHARACTERISTIC DOES NOT INTERSECT THE SHOCK
     1")
C 403 FORMAT(1H ,"  XA   = ",E15.8,"  DT2   = ",E15.8)
C 500 FORMAT(1H ,"POINT   A  VARIABLES FOLLOW ")
C 501 FORMAT(1H ,"POINT   B  VARIABLES FOLLOW ")
```

```
C 502 FORMAT (1H ," X U C P RH E ")
  510 FORMAT(" XB NOT CONVERGE IN GNPT")
  520 FORMAT(" XA NOT CONVERGE IN GNPT")
C 530 FORMAT(" FROM GNPT WE HAVE U3 =", E15.8, " DU4 =", E15.8,
C    1  " DX4 =", E15.8, " X3 =", E15.8, /, " C1 =", E15.8, "
C        DC4 =",
C    2  E15.8, " X4 =", E15.8)
C   -    KQ = 1;  GENERAL POINT ADJACENT TO LEFT RUNNING SHOCK.
C   -             FIRST CHARACTERISTIC INTERSECT WITH SHOCK.
C   -   KQ = 2;  GENERAL POINT ADJACENT TO RIGHT RUNNING SHOCK.
C   -             SECOND CHARACTERISTIC INTERSECT WITH SHOCK.
C   -    KQ = 3;  REGULAR GENERAL POINT.
      LINP ( V1, DV, DX, DY) = V1 + DV *DY / DX
      QINP( V2, DV1, DV2, DY) = V2 + (DV2*DX1**2+DV1*DX2**2)* DY
     1/ (       DX1*DX2*(DX1 + DX2)) + (-DV1*DX2 + DV2*DX1)*(  DY
     2 )**2/(DX1*DX2   *(DX1 + DX2))
      NIT = 20
      MNIT = 4
      TOL1 = 1.E-10
      TOL = 0.0005
      KP = KQ
      GO TO (201, 202, 200), KQ
  201 PRINT 300
      GO TO 200
  202 PRINT 301
  200 CONTINUE
C     PRINT 502
      U1 = U(1, I1)
      E1 = E(1, I1)
      RH1 = RH(1, I1)
      Q1 = Q(1, I1)
      C1 = C(1, I1)
      P1 = P(1, I1)
      U2 =U(1, I2)
      E2 = E(1, I2)
      RH2 = RH(1, I2)
      Q2 = Q(1, I2)
      P2 = P(1, I2)
      X1 = X(1, I1)
      X2 = X(1, I2)
      X3 = X(1, I3)
      C3 = C(1, I3)
      U3 = U (1,I3)
      E3 = E(1, I3)
      P3 = P(1, I3)
      Q3 = Q(1, I3)
      K = 1
      KM1 = 3
      U4 = U(1, I2)
      DX1 = X(1, I2) - X(1, I1)
```

```
      DU1 = U(1, I2) - U(1, I1)
      DC1 = C(1, I2) - C(1, I1)
      DE1 = E(1, I2) - E(1, I1)
      DRH1 = RH(1, I2) - RH(1, I1)
      DP1 = P(1, I2) - P(1, I1)
      DQ1 = Q(1, I2) - Q(1, I1)
      DX2 = X(1, I3) - X(1, I2)
      DC2=C(1,I3)-C(1,I2)
      DU2 = U(1, I3) - U(1, I2)
      DE2 = E(1, I3) - E(1, I2)
C THIS IS THE POINT OF THE INFAMOUS COBOL FOULUP
      DP2 = P(1, I3) - P(1, I2)
      DRH2 = RH(1, I3) - RH(1, I2)
      DQ2 = Q(1, I3) - Q(1, I2)
      GO TO (2, 4, 10), KQ
C   -   B LIES ON THE LEFT SHOCK.
    2 DX3 = X(1, I1) - X(2, I1)
      DU3 = U(1, I1) - U(2, I1)
      DE3 = E(1, I1) - E(2, I1)
      DP3 = P(1, I1) - P(2, I1)
      DRH3 = RH(1, I1) - RH(2, I1)
      DQ3 = Q(1, I1) - Q(2, I1)
      DC3 = C(1, I1) - C(2, I1)
      X1P = X(2,I1)
      X4 = X2 + U2*DT
      DT1 = DT/2.
      XB = (X1 + X1P)/2.
      KCOUNT = 1
      B1 = U1+C1-(X1+X1P)/DX3*(DU3+DC3)+DX3/DT
      B2=X1P*(U1+C1-X1*(DU3+DC3)/DX3+X4/X1P*DX3/DT)
      B3=- 1./DX3*(DU3+DC3)
    5 CONTINUE
      FP=1.-2.*B3*XB/B1
      F = XB - ( B2 + B3 * XB **2)/B1
      XBP=XB-F/FP
      IF ( ABS((XBP - XB)/XBP)  .LT. TOL) GO TO 6
      KCOUNT = KCOUNT + 1
      IF(KCOUNT .GT. NIT) GO TO 7
      XB= XBP
C     PRINT 400,XB,DT1
      GO TO 5
    7 PRINT 510
C     PRINT 400, XB, DT1
    6 CONTINUE
      DT1P = DT/(X1P - X1) * ( X1P -XB)
      DT1 = DT1P
      IF(DT1 .GT. 1.0*DT) GO TO 3
      KM1 = 1
      GO TO 10
    3 KM1 = 3
```

```
      PRINT 401
      GO TO 10
C     -   A LIES ON THE RIGHT SHOCK.
    4 DX4 = X(1, I3) - X(2, I3)
      DU4 = U(1, I3) - U(2, I3)
      DE4 = E(1, I3) - E(2, I3)
      DP4 = P(1, I3) - P(2, I3)
      DRH4 = RH(1, I3) - RH(2, I3)
      DQ4 = Q(1, I3) - Q(2, I3)
      DC4 = C(1, I3) - C(2, I3)
      X3P = X(2,I3)
      X4 = X2 + U2*DT
      XA = (X3P + X3)/2.
      DT2 = DT / 2.
C     PRINT 530, U3, DU4, DX4, X3, C3, DC4, X4
      KCOUNT = 1
      A1 = U3-C3-(X3+X3P)/DX4*(DU4-DC4)+DX4/DT
      A2 = X3P*(U3-C3-X3*(DU4-DC4)/DX4+X4/X3P*DX4/DT)
      A3 = - 1./DX4*(DU4-DC4)
    9 CONTINUE
      FP = 1.-2.*A3*XA/A1
      F = XA - ( A2+A3*XA**2)/A1
      XAP = XA - F/FP
      IF ( ABS((XAP - XA)/XAP ) .LT. TOL ) GO TO 11
      KCOUNT = KCOUNT + 1
      IF(KCOUNT .GT. NIT) GO TO 12
      XA = XAP
C     PRINT 403, XA, DT2
      GO TO 9
   12 PRINT 520
C     PRINT 403, XA, DT2
   11 CONTINUE
      DT2P = DT / ( X3P -X3) * ( X3P -XA)
      DT2 = DT2P
      IF(DT2 .GT. 1.0*DT) GO TO 8
      KM1 = 2
      GO TO 10
    8 KM1 = 3
      PRINT 402
C     -   BEGINNING OF ITERATION LOOP.
   10 CONTINUE
      X4 = X2 + U2 * DT
      KP = KM1
      GO TO (15, 20, 25), KP
   15 CONTINUE
      DT2 = DT
      DXB = XB - X1
      UB = LINP(U1, DU3, DX3, DXB)
      EB = LINP(E1, DE3, DX3, DXB)
      PB = LINP(P1, DP3, DX3, DXB)
```

```
      RHB = EQSTRQ(L, EB, PB)
      CB = EQSTCQ(L, EB, RHB, PB)
      QB = LINP(Q1, DQ3, DX3, DXB)
      PEB = EQSTPE(L,EB,RHB)
      GO TO 30
   20 CONTINUE
      DT1 = DT
      DXA = XA - X3
      UA = LINP(U3, DU4, DX4, DXA)
      EA = LINP(E3, DE4, DX4, DXA)
      PA = LINP(P3, DP4, DX4, DXA)
      RHA = EQSTRQ(L, EA, PA)
      CA = EQSTCQ(L, EA, RHA, PA)
      QA = LINP(Q3, DQ4, DX4, DXA)
      PEA = EQSTPE(L,EA,RHA)
      GO TO 28
   25 DT1 = DT
      DT2 = DT
   28 CONTINUE
      IF ( K .NE. 1 ) GO TO 26
      XB=(X4-DT*(U1+C1-X1/DX1*(DU1+DC1)))/(1.0+DT/DX1*(DU1+DC1))
      GO TO 29
   26 CONTINUE
   27 DXB = XB - X2
      UB = QIN P(U2, DU1, DU2,DXB)
      EB = QIN P(E2, DE1, DE2,DXB)
      QB = QIN P(Q2, DQ1, DQ2,DXB)
      PB = QINP(P2, DP1, DP2, DXB)
      RHB = EQSTRQ(L, EB, PB)
      CB = EQSTCQ(L, EB, RHB, PB)
      PEB = EQSTPE(L, EB, RHB)
       IF ( KP .EQ. 3) GO TO 30
      GO TO 40
   29 CONTINUE
      DXB = XB - X1
      UB = LINP(U1, DU1, DX1,DXB)
      EB = LINP(E1, DE1, DX1,DXB)
      QB = LINP(Q1, DQ1, DX1,DXB)
      PB = LINP(P1, DP1, DX1, DXB)
      RHB = EQSTRQ(L, EB, PB)
      CB = EQSTCQ(L, EB, RHB, PB)
      PEB = EQSTPE(L, EB, RHB)
      IF(KP .EQ. 3) GO TO 30
      GO TO 40
   30 CONTINUE
      IF ( K .NE. 1) GO TO 32
      XA=(X4-DT*(U3-C3-X3/DX2*(DU2-DC2)))/(1.0+DT/DX2*(DU2-DC2))
      GO TO 31
   32 CONTINUE
   33 DXA = XA -X2
```

```
      UA = QIN P(U2, DU1, DU2,DXA)
      EA = QIN P(E2, DE1, DE2,DXA)
      QA = QIN P(Q2, DQ1, DQ2,DXA)
      PA = QINP(P2, DP1, DP2, DXA)
      RHA = EQSTRQ(L, EA, PA)
      CA = EQSTCQ(L, EA, RHA, PA)
      PEA = EQSTPE(L, EA, RHA)
      GO TO 40
   31 CONTINUE
      DXA = XA - X2
      UA = LIN P ( U2 , DU2, DX2,DXA )
      EA = LIN P ( E2, DE2, DX2, DXA )
      QA = LIN P ( Q2, DQ2, DX2, DXA )
      PA = LINP(P2, DP2, DX2, DXA)
      RHA = EQSTRQ(L, EA, PA)
      CA = EQSTCQ(L, EA, RHA, PA)
      PEA = EQSTPE(L, EA, RHA)
   40 CONTINUE
      IF ( K .NE. 1 ) GO TO 35
      Q4 = Q2
      RHC4A = RHA *CA
      RHC4B = RHB * CB
      UCX4B = UB*CB/XB
      UCX4A = UA*CA/XA
      P RCQ4B = PEB*QB/(RHB*CB)
      P RCQ4A = PEA*QA/(RHA*CA)
      PEQ4A = PEA*QA
      PEQ4B = PEB*QB
      GO TO 41
   35 RHC4A = (RH4*C4 + RHA*CA)/2.
      RHC4B = (RH4*C4 + RHB*CB)/2.
      UCX4B = (U4*C4/X4 + UB*CB/XB)/2.
      UCX4A = (U4*C4/X4 + UA*CA/XA)/2.
      P RCQ4B = (PEB *QB/(RHB*CB) + PE4*Q4/(RH4*C4))/2.
      P RCQ4A = (PEA*QA/(RHA*CA) + PE4*Q4/(RH4*C4))/2.
      PEQ4A = (PEA*QA + PE4*Q4)/2.
      PEQ4B = (PEB*QB + PE4*Q4)/2.
   41 CONTINUE
C     PRINT 500
C     PRINT 103,I,I2,XA,UA,CA,PA,RHA,EA,L
C     PRINT 501
C     PRINT 103,I,I2,XB,UB,CB,PB,RHB,EB,L
      TOLCON = 0.0005
      P4P = (PB/RHC4B + PA/RHC4A + UB - UA + (-(XMU - 1.)*UCX4B +
     1   P RCQ4B)*DT1 + (-(XMU - 1.)*UCX4A + P RCQ4A)*DT2)/(1./RH
     1C4B +       1./RHC4A)
      U4P = (PB - PA + RHC4B*UB + RHC4A*UA + (-(XMU - 1.)*UCX4B*R
     1HC4B +     PEQ4B)*DT1 - (-(XMU - 1.)*UCX4A*RHC4A + PEQ4A)*DT
     22)/(RHC4A +       RHC4B)
      IF ( K .NE. 1 ) GO TO 43
```

```
      RH4 = RH2
   45 E4P = E2 + P2/(RH2**2)*(RH4 - RH2) + Q2*DT
      RH4P = EQSTRQ(L, E4P, P4P)
      IF ( ABS (( RH4 - RH4P )/RH4) .LT. TOLCON) GO TO 46
C 528 FORMAT(1H ," RH4=",E15.8,"    RH4P=",E15.8)
C     PRINT 528,RH4,RH4P
      RH4 = (RH4P + RH4) / 2.0
      GO TO 45
   43 E4P = E2 + (P2 + P4P)/(RH4**2 + RH2**2)*(RH4 - RH2) + (Q4 +
     1 Q2)/      2.*DT
      RH4P = EQSTRQ(L, E4P, P4P)
   46 C4P = EQSTCQ(L, E4P, RH4P, P4P)
      IF ( K .EQ. 1) GO TO 50
      IF ( ABS ((P4P -P4) /P4P) - TOLCON) 42, 42, 50
   42 IF (ABS(U4P).LT. TOL1) GO TO 44
      IF ( ABS (( U4P - U4)/U4P) - TOLCON) 44, 44, 50
   44 IF ( ABS ((E4P - E4)/ E4P) - TOLCON) 70, 70, 50
   50 IF ( K .GE. NIT ) GO TO 60
      IF ( K .EQ. MNIT ) GO TO 70
C     PRINT 103,I,I2,X4,U4P,C4P,P4P,RH4P,E4P,L
      K = K + 1
      IF(K .GE. 3) GO TO 55
      U4 = U4P
      C4 = C4P
      P4 = P4P
      RH4 = RH4P
      E4 = E4P
      PE4 = EQSTPE (L,E4,RH4)
      GO TO 10
   55 U4 = (U4 + U4P)/2.
      C4 = (C4 + C4P)/2.
      P4 = (P4 + P4P)/2.
      RH4 = (RH4 + RH4P) /2.
      E4 = (E4 + E4P)/2.
      PE4 = EQSTPE(L, E4, RH4)
      GO TO 10
C   -   END OF ITERATION LOOP.
   60 PRINT 1
   70 X(2, I2) = X4
      U(2, I2) = U4P
       IF (P4P .LE. 0 ) P4P=-P4P
       IF (RH4P .LE. 0) RH4P= -RH4P
      C(2, I2) = C4P
      RH(2, I2) = RH4P
      E(2, I2) = E4P
      P(2, I2) = P4P
      Q(2, I2) = Q(1, I2)
      RETURN
      END
```

```
      SUBROUTINE INIDAT
      DIMENSION A(40)
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/INIT/PI1,UI1,RHI1,EI1,CI1,QI1,PI2,UI2,RHI2,EI2,CI2,Q
     1I2
      COMMON/SINPT/ PI, UI, RHI, UF, XZ, PF, IR1, IR2
      COMMON/RAWAV / XR(30),UR(30),CR(30),RHR(30),ER(30),PR(30),R
     1HP(30)
      COMMON /C1AND2/ PF1,UF1,RHF1,EF1,CF1,QF1,PF2,UF2,RHF2,EF2,C
     1F2,QF2,   XF,XS
      COMMON/DTTS/DTT
      COMMON/NDIM/TT,TTMAX,XXZ,TMAX
      COMMON/REFL/TREF,T
      COMMON/GAM/GAMM(2)
      COMMON/CON/TCON,DT2,PTOL
      COMMON/NNEWW/KSHOCK,KTLLG,IDT
  905 FORMAT(5E15.8)
  907 FORMAT(1H ,"INITIAL TIME= 0.0"/
     (1H ,"(T)TIME OF THE FIRST TIME LINE          ",1PE15.8/
     (1H ,"(TMAX)MAXIMUM RUN TIME                  ",1PE15.8)
  908 FORMAT(1H0,"GAMMA FOR REGION ONE",1PE15.8/
     (1H ,"GAMMA FOR REGION TWO",1PE15.8)
  911 FORMAT(1H ,"REGION TWO PROPERTIES "/
     (1H ,6X,"UI2",12X,"CI2",12X,"PI2",11X,"RHI2",12X,"EI2"/
     (1H ,5(1PE15.8))
  915 FORMAT(35A2)
  916 FORMAT(I2,3F15.8)
  917 FORMAT(1H1,"DIMENSIONAL INPUT DATA"////
     (1H ,"TIME UNITS ARE(",5A2,")")
  918 FORMAT(1H0,5(2X,"("5A2,")")       )
  919 FORMAT(1H ,"REGION ONE PROPERTIES"/
     (1H ,6X,"UI1",12X,"CI1",12X,"PI1",11X,"RHI1",12X,"EI1"/
     (1H ,5(1PE15.8))
  920 FORMAT(1H0,"(IN)THE INITIAL RAREFACTION IS DIVIDED INTO ",I
     14,      " SUBDIVISIONS"/
     2                1H ,"(XZ)RADIUS OF REGION ONE ",1PE15.8,"(",5
     3A2,")")
  921 FORMAT(1H0,"(XZ)RADIUS OF REGION ONE ",1PE15.8)
  922 FORMAT(////" NON-DIMENSIONAL INPUT DATA"///)
  925 FORMAT(1H0,"(PTOL)SHOCK PRESSURE JUMP TOLERANCE ",1PE15.8)
      READ 905,T
      DT=T
      READ 905,GAMM(1),GAMM(2)
      READ 915,(A(J),J=1,35)
      QI1=0.0
      QI2=0.0
      READ 905,PI1,UI1,RHI1
      READ 905,PI2,UI2,RHI2
      READ 916,IN,XZ,PTOL
      EI1=EQSTEQ(1,RHI1,PI1)
```

```
      CI1=EQSTCQ(1,EI1,RHI1,PI1)
      EI2=EQSTEQ(2,RHI2,PI2)
      CI2=EQSTCQ(2,EI2,RHI2,PI2)
      PRINT 917,(A(J),J=1,5)
      PRINT 907,T,TMAX
      PRINT 908,GAMM(1),GAMM(2)
      PRINT 918,(A(J),J=6,30)
      PRINT 919,UI1,CI1,PI1,RHI1,EI1
      PRINT 911,UI2,CI2,PI2,RHI2,EI2
      PRINT 920,IN,XZ,(A(J),J=31,35)
      PRINT 925,PTOL
      XXZ=XZ
      XZ=1.0
      TT=T
      DTT=DT
      TTMAX=TMAX
      A(1)=XXZ
      A(2)=XXZ/CI1
      A(3)=CI1
      A(4)=CI1
      A(5)=RHI1*CI1**2
      A(6)=RHI1
      A(7)=CI1**2
      CALL NONDIM
      PRINT 922
      T=TT
      DT=DTT
      TMAX=TTMAX
      PTOL=PTOL*XZ
      PRINT 907,T,TMAX
      PRINT 919,UI1,CI1,PI1,RHI1,EI1
      PRINT 911,UI2,CI2,PI2,RHI2,EI2
      PRINT 921,XZ
      PRINT 925,PTOL
      PRINT 923,(A(J),J=1,7)
  923 FORMAT(//1H ,"CONVERSION FACTORS BACK TO DIMENSIONAL QUANTI
     1TIES"/ 1H0,"QUANTITY MULTIPLY BY"/
     2                    1H ,"X",8X,1PE15.8/
     3                                 1H ,"T",8X,1PE15.8/
     4                                              1H ,"U",8X,1PE15.8/
     5                                                          1H ,"C",8X,1PE15.8/
     6                                                                      1H ,"P",8X,1PE15.
     78/                                                                                1H ,"RH",7
     8X,1PE15.8/                                                                                    1H
     9,"E",8X,1PE15.8)
      TCON=A(2)
      CALL RASHOK(2,IN,1)
      CALL ASIGPT
      IF(IDT.EQ.1)DT=DT2/TCON
      RETURN
```

```
      END


      SUBROUTINE INTFPT(I1,I2,I3,I4)
C      SUBROUTINE TO CALCULATE PROPERTIES AT INTERFACE.
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
  100 FORMAT(1H ,I2,1X,I4,3X,6(E15.8,2X),"*",I1,2X,"INTF")
  110 FORMAT(1H ,10X,"INTERFACE SOLUTION DIVERGENT")
C 120 FORMAT("  *****", 6(E15.8, 2X), "*****")
C2000 FORMAT(1H ,5X,"XA",10X,"UA",10X,"PA",10X,"RHA",9X,"EA",/,
C        1H ,5E11.
C     1   4)
C2001 FORMAT(1H ,5X,"XB",10X,"UB",10X,"PB",10X,"RHB",9X,"EB",/,
C        1H ,5E11.
C     $   4)
      XINP(V1,DV,DX,Y)=V1+DV*Y/DX
      L = 1
      TOL=0.0005
      TOL1 = 1.E-10
      LIM=30
      K=1
C    -  DEFINE PROPERTIES AT POINTS 1, 2, 3, AND 4.
      U1 = U(1, I1)
      C1 = C(1, I1)
      X1 = X(1, I1)
      RH1 = RH(1, I1)
      E1 = E(1, I1)
      P1 = P(1, I1)
      Q1 = Q(1, I1)
      E2 = E(1, I2)
      P2 = P(1, I2)
      RH2 = RH(1, I2)
      Q2 = Q(1, I2)
      C2 = C(1, I2)
      U2 = U(1, I2)
      X3 = X(1, I3)
      U3 = U(1, I3)
      C3 = C(1, I3)
      RH3 = RH(1, I3)
      E3 = E(1, I3)
      P3 = P(1, I3)
      X4 = X(1, I4)
      Q3 = Q(1, I3)
      U4 = U(1, I4)
      C4 = C(1, I4)
C      DEFINE DIFF. QUANTITIES
      DX2=X(1,I4)-X(1,I3)
```

```
      DU2=U(1,I4)-U(1,I3)
      DC2 = C(1, I4) - C(1, I3)
      DE2=E(1,I4)-E(1,I3)
      DRH2=RH(1,I4)-RH(1,I3)
      DP2 = P(1, I4) - P(1, I3)
      DQ2=Q(1,I4) -Q(1,I3)
      DX1=X(1,I2)-X(1,I1)
      DU1=U(1,I2)-U(1,I1)
      DE1=E(1,I2)-E(1,I1)
      DRH1 = RH(1,I2) - RH(1,I1)
      DP1 = P(1, I2) - P(1, I1)
      DQ1=Q(1,I2)-Q(1,I1)
      DC1 = C2 -C1
      XA = (X4A - (U4 - C4 - (DU2 - DC2)*X4/DX2)*DT)/(1. + (DU2 -
     1 DC2)*    DT/DX2)
      X4A = X3 + U3*DT
C  -   ESTIMATE PTS A AND B
      XB = (X4A - (U1 + C1 - (DU1 + DC1)*X1/DX1)*DT)/(1. + (DU1+
     1DC1)*DT   /DX1)
      DXA = XA - X3
      DXB = XB - X1
      UA = XINP(U3, DU2, DX2, DXA)
      CA = XINP(C3, DC2, DX2, DXA)
      PA = XINP(P3, DP2, DX2, DXA)
      QA = XINP(Q3, DQ2, DX2, DXA)
      EA = XINP(E3, DE2, DX2, DXA)
      RHA = EQSTRQ(L, EA, PA)
      UB = XINP(U1, DU1, DX1, DXB)
      CB = XINP(C1, DC1, DX1, DXB)
      PB = XINP(P1, DP1, DX1, DXB)
      QB = XINP(Q1, DQ1, DX1, DXB)
      EB = XINP(E1, DE1, DX1, DXB)
      RHB = EQSTRQ(L, EB, PB)
C  -   ASSUME VALUES FOR VARIABLES
      U4A = U3
      U4B = U2
      C4A = CA
      RH4A = RH3
      E4A = E3
      P4A = P3
      P4B = P4A
      C4B = CB
      RH4B = RH2
      E4B = E2
      Q34A = Q3
      Q24B = Q2
    1 CONTINUE
C      BEGINNING OF ITERATION LOOP
      X4A = X3 + U3*DT
      X4B=X4A
```

```
      UMC4A=(U4A+UA-C4A-CA)/2.
      UPC4B=(U4B+UB+C4B+CB)/2.
      XA=X4A-UMC4A*DT
      XB=X4B-UPC4B*DT
      DXA=XA-X3
      DXB=XB-X1
      UA=XINP(U3,DU2,DX2,DXA)
      EA=XINP(E3,DE2,DX2,DXA)
      PA = XINP(P3, DP2, DX2, DXA)
      RHA = EQSTRQ(2, EA, PA)
      CA = EQSTCQ(2, EA, RHA, PA)
      PEA = EQSTPE(2, EA, RHA)
      PE4A = EQSTPE(2, E4A, RH4A)
      QA=XINP(Q3,DQ2,DX2,DXA)
      UB=XINP(U1,DU1,DX1,DXB)
      EB=XINP(E1,DE1,DX1,DXB)
      PB = XINP(P1, DP1, DX1, DXB)
      RHB = EQSTRQ(1, EB, PB)
      CB = EQSTCQ(1, EB, RHB, PB)
      PEB = EQSTPE(1, EB, RHB)
      PE4B = EQSTPE(1, E4B, RH4B)
      QB=XINP(Q1,DQ1,DX1,DXB)
      Q4A = Q3
      Q4B = Q2
      RHC4B = (RHB*CB + RH4B*C4B)/2.
      RHC4A = (RHA*CA + RH4A*C4A)/2.
      UCX4B = (UB*CB/XB + U4B*C4B/X4B)/2.
      UCX4A = (UA*CA/XA + U4A*C4A/X4A)/2.
      P RCQ4B = (PEB*QB/(RHB*CB) + PE4B*Q4B/(RH4B*C4B))/2.
      P RCQ4A = (PEA*QA/(RHA*CA) + PE4A*Q4A/(RH4A*C4A))/2.
      PEQ4B = (PEB*QB + PE4B*Q4B)/2.
      PEQ4A = (PEA*QA + PE4A*Q4A)/2.
   10 CONTINUE
      PRH34A = (P3/(RH3**2)+P4A/(RH4A**2))/2.
      PRH24B = (P2/(RH2**2)+P4B/(RH4B**2))/2.
      P4AP = (PB/RHC4B + PA/RHC4A + UB - UA + (-(XMU - 1.)*(UCX4B
     1 +          UCX4A) + P RCQ4B + P RCQ4A)*DT)/(1./RHC4B + 1./RH
     2C4A)
      P4BP = P4AP
C     PRINT 2000,XA,UA,PA,RHA,EA
C     PRINT 2001,XB,UB,PB,RHB,EB
      U4AP = (PB - PA + RHC4B*UB + RHC4A*UA + (-(XMU - 1.)*UCX4B*
     1RHC4B +    PEQ4B + (XMU - 1.)*UCX4A*RHC4A - PEQ4A)*DT)/(RHC4
     2A + RHC4B)
      E4AP = E3 + PRH34A*(RH4A - RH3) + Q34A*DT
      E4BP = E2 + PRH24B*(RH4B - RH2) + Q24B*DT
      RH4AP = EQSTRQ(2, E4AP, P4AP)
      RH4BP = EQSTRQ(1, E4BP, P4BP)
      C4AP = EQSTCQ(2, E4AP, RH4AP, P4AP)
      C4BP = EQSTCQ(1, E4BP, RH4BP, P4BP)
```

```
      IF( ABS((P4AP - P4A)/P4A) .GT. TOL) GO TO 3
      IF( ABS(U4A) .LT. TOL1) GO TO 40
      IF( ABS((U4AP - U4A)/U4A) .GT. TOL) GO TO 3
   40 IF( ABS((E4AP - E4A)/E4A) .GT. TOL) GO TO 3
      IF( ABS((E4BP - E4B)/E4B) .GT. TOL) GO TO 3
      IF( ABS((RH4AP - RH4A)/RH4A) .GT. TOL) GO TO 3
      IF( ABS((RH4BP - RH4B)/RH4B)  -   TOL) 6, 6, 3
    3 IF(K .LT. LIM) GO TO 7
      PRINT 110
      GO TO 6
    7 U4A=(U4AP+U4A)/2.
      U4B=U4A
      C4A=(C4AP+C4A)/2.
      C4B=(C4BP+C4B)/2.
      P4A=(P4AP+P4A)/2.
      P4B = P4A
    4 RH4A=(RH4A+RH4AP)/2.
      RH4B=(RH4B+RH4BP)/2.
       E4A=(E4A+E4AP)/2.
      E4B=(E4B+E4BP)/2.
      L = 2
C     PRINT 100, I,I2,X4A,U4A,C4A,P4A,RH4A,E4A,L
      L = 1
C     PRINT 100, I,I3,X4B,U4B,C4B,P4B,RH4B,E4B,L
    5 K = K + 1
      GO TO 1
C      END OF ITERATION LOOP
    6 U(2,I2)=U4AP
      U(2,I3)=U4AP
        P(2,I2)=P4AP
      P(2,I3)=P4AP
      C(2,I2)=C4BP
      C(2,I3)=C4AP
      E(2,I2)=E4BP
      E(2,I3)=E4AP
      RH(2,I2)=RH4BP
      RH(2,I3)=RH4AP
      Q(2, I2) = 0.
      Q(2, I3) = 0.
      X(2,I2)=X A
      X(2,I3)=X4B
      RETURN
      END


      SUBROUTINE NONDIM
      COMMON/INIT/PI1,UI1,RHI1,EI1,CI1,QI1,PI2,UI2,RHI2,EI2,CI2,Q
     1I2
      COMMON/NOM/PO,RHO,EO,CO
```

```
      COMMON/DTTS/DTT
      COMMON/NDIM/TT,TTMAX,XXZ
      PO=PI1
      RHO=RHI1
      EO=EI1
      CO=CI1
      TT=TT*CO/XXZ
      DTT=DTT*CO/XXZ
      TTMAX=TTMAX*CO/XXZ
      PI1=PI1/(RHO*CO**2)
      RHI1 = RHI1/RHO
      UI1 = UI1/CO
      CI1 = CI1/CO
      EI1=EI1/CO**2
      PI2=PI2/(RHO*CO**2)
      RHI2 = RHI2/RHO
      UI2 = UI2/CO
      CI2 = CI2/CO
      EI2=EI2/CO**2
      RETURN
      END


      SUBROUTINE PTARNG
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),    E(2,1000),P(2,1000)
      COMMON /TIMUU/ DT,UU1,UU2,I,XMU
      COMMON/NOCON/ IS1, IS2, IS3, IS4, INT1, INT2, IFF
      COMMON/DTTS/ DTT
      COMMON/NNEWW/KSHOCK
      DIMENSION ME(1000), MA(1000)
 800  FORMAT(1H ,I2,1X,I4,3X,6(E15.8,2X),I2)
  801 FORMAT (1H ,10X,I4," POINTS BEING ELIMINATED")
  802 FORMAT(1H ,10X,I4," POINTS BEING ADDED")
  810 FORMAT(11X, "  POINT ELIMINATED AT (1,", I4, ")")
  812 FORMAT(11X, " NO POINT BEING ADDED.")
  814 FORMAT(11X, " NEGATIVE NUMBER OF POINTS ADDED.")
C 815 FORMAT(11X, I4, " POINTS BETWEEN KK =", I4, " AND KK =",
C        I4,
C     1  " BEING ADDED.")
  820 FORMAT(11X, " POINT AT KK =", I4, " BEING ELIMINATED DUE TO
     1 INTERSECTION WITH SHOCK WAVE")
  825 FORMAT ( 1H ,10X,I4,"POINTS ADDED BET. KK = ",I4," AND KK =
     1 ",I4)
  830 FORMAT("     K =", I5, " AND IN =", I5)
C 840 FORMAT("  *****")
C 841 FORMAT("  ******")
C 842 FORMAT("  *******")
C 843 FORMAT("  ********")
```

```
C 850 FORMAT("   DTMAX =", E15.8, "   TE =", E15.8)
  860 FORMAT("   NN =", I4, 4(E15.8, 2X))
 1000 FORMAT("   IS1 =", I5, " IS2 =",I5, " INT1 =",I5, " INT2 =",
     1 I5,     "   IS3 =", I5, "   IS4 =", I5, "   IMAX =", I5)
C1050 FORMAT(4E15.8)
C1060 FORMAT(  "   E(1,J) =",E15.8, " P(1,J) =",E15.8," RH(1,J)
C          =",E15.8,
C     1  " X(1,J) =",E15.8," J =",I5)
C1100 FORMAT("   J1 =", I5, 5X, " J =", I5, 5X, " DT =", E15.8)
C1110 FORMAT("   DT =", E15.8)
C1200 FORMAT("   IFF =", I5, 10X, " DT =", E15.8)
C1300 FORMAT("   C(1,J) =", E15.8, " E(1,J) =",E15.8," P(1,J) =",
C          E15.8,
C     1  " RH(1,J) =",E15.8," X(1,J) =",E15.8," J =",I5)
      XLINP(V1, DV, DX, DY) = V1 + DV*DY/DX
      QINP( V2, DV1, DV2, DY) = V2 + (DV2*DX1**2+DV1*DX2**2)* DY
     1/ (        DX1*DX2*(DX1 + DX2)) + (-DV1*DX2 + DV2*DX1)*(  DY
     2 )**2/(DX1*DX2   *(DX1 + DX2))
C          IF(I .GE.  3)  DT1=DTT
      IF(I .EQ.1) DT1=DT
      IF(DT .GT. DT1) DT1=DT
C     DTT = .005
C     DT1 = DT*6./5.
C     IF(I .EQ. 1) DT1 = DT
C     IF(DT1 .GE. DTT) DT1 = DTT
      TMIN = 0.8
      TMAX = 2.0
      TMAX = 2.0
      TMIN = 0.5
      TMAX = 100.0
        TMAX=5.0
      TMIN = 0.9
      TMAX = 3.0
      DTMIN = TMIN*DT1
      DTMAX = TMAX*DT1
 301  FORMAT(1H ,"DTMIN=",E15.8,"DTMAX=",E15.8)
      IFF1 = IFF
      IK = 0
      IS1M1 = IS1 - 1
      KKM1 = 1
      IS1M2 = IS1 - 2
      IF(KSHOCK.EQ.1) GO TO 400
      IF(IS1M2 .LE. 1) IS1P = IS1
      IF(IS1M2 .LE. 1) GO TO 15
      DO 10 KK = 2, IS1M1
      IF(IK .EQ. (IS1 - 3)) GO TO 17
    9 TE = ADT(KK, KKM1)
      IF(TE .GT. DTMIN) GO TO 8
      IK = IK + 1
      ME(IK) = KK
```

```
      GO TO 10
    8 KKM1 = KK
   10 CONTINUE
      IS1P = IS1 - IK
      IF (IK .EQ. 0) GO TO 11
      IF(ME(IK) .EQ. IS1M1) GO TO 15
   11 TE = ADT(IS1, IS1M1)
      IF(TE .GT. DTMIN) GO TO 15
      IK = IK + 1
      ME(IK) = IS1M1
   17 CONTINUE
      IS1P = IS1 - IK
   15 CONTINUE
      IS2P1 = IS2 + 1
      INT1M1 = INT1 - 1
      IF(IS2P1 .EQ. INT1) GO TO 24
      KKM1 = IS2
      GO TO 401
  400 IS2P1=2
      INT1M1=INT1-1
      IS1P=IS1
401   CONTINUE
      DO 20 KK = IS2P1, INT1M1
   16 TE = ADT(KK, KKM1)
      IF(TE .GT. DTMIN) GO TO 19
      IK = IK + 1
      ME(IK) = KK
      GO TO 20
   19 KKM1 = KK
   20 CONTINUE
   22 CONTINUE
      INT1P = INT1 - IK
      IF ( IK .EQ. 0) GO TO 23
      IF(ME(IK) .EQ. INT1M1) GO TO 25
   23 TE = ADT(INT1, INT1M1)
      IF(TE .GT. DTMIN) GO TO 25
      IK = IK + 1
      ME(IK) = INT1M1
   24 INT1P = INT1 - IK
   25 CONTINUE
      INT2P1 = INT2 + 1
      IS3M1 = IS3 - 1
      IF(INT2P1 .EQ. IS3) GO TO 32
      KKM1 = INT2
       IF ( INT2P1.EQ. IS3M1 ) GO TO  31
      DO 30 KK = INT2P1, IS3M1
   26 TE = ADT(KK, KKM1)
      IF(TE .GT. DTMIN) GO TO 29
      IK = IK + 1
      ME(IK) = KK
```

```
      GO TO 30
   29 KKM1 = KK
   30 CONTINUE
   31 CONTINUE
      IS3P = IS3 - IK
      IF ( IK .EQ. 0 ) GO TO 34
      IF(ME(IK) .EQ. IS3M1) GO TO 35
   34 TE = ADT(IS3, IS3M1)
      IF(TE .GT. DTMIN) GO TO 35
      IK = IK + 1
      ME(IK) = IS3M1
   32 IS3P = IS3 - IK
   35 CONTINUE
      IS4P1 = IS4 + 1
      IFFM1 = IFF - 1
      KKM1 = IS4
      DO 40 KK = IS4P1, IFFM1
   36 TE = ADT(KK, KKM1)
      IF(TE .GT. DTMIN) GO TO 39
      IK = IK + 1
      ME(IK) = KK
      GO TO 40
   39 KKM1 = KK
   40 CONTINUE
      IFFP = IFF - IK
      IF(IK .EQ. 0) GO TO 41
      IF(ME(IK) .EQ. IFFM1) GO TO 45
   41 TE = ADT(IFF, IFFM1)
      IF(TE .GT. DTMIN) GO TO 45
      IK = IK + 1
      ME(IK) = IFFM1
      IFFP = IFF - IK
   45 CONTINUE
      IS1 = IS1P
C     -   END OF ELIMINATING POINTS.
      IS2 = IS1P + 1
      IS3 = IS3P
      IS4 = IS3P + 1
      INT2 = INT1P + 1
      INT1 = INT1P
      IFF = IFFP
      IF ( IK .EQ. 0)  GO TO 56
      JK = 1
      K = 1
      DO 55 KK = 1,IFF1
      IF ( ME(JK) .EQ. KK ) GO TO 53
      P( 1, K ) = P( 1 , KK)
      U ( 1,K) = U(1,KK)
      RH(1, K ) = RH(1,KK )
      E(1,K) = E(1,KK)
```

```
      C(1,K) = C(1,KK)
      Q(1,K) = Q(1,KK)
      X(1,K) = X(1,KK)
      K = K + 1
      GO TO 55
   53 CONTINUE
      IF ( JK .EQ. IK) GO TO 55
      JK = JK + 1
   55 CONTINUE
   56 CONTINUE
      IF(IK .EQ. 0) GO TO 49
      DO 48 J = 1, IK
 48   CONTINUE
   49 CONTINUE
C     DO 50 J = 1,IFF
C  50 PRINT 800, I,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J),L
C   -   ADD POINTS STARTS.
C   -   IK = TOTAL NUMBER OF POINTS TO BE ADDED.
C      IF(IFF .GT. 0 ) GO TO 100
      IK = 0
      IF(KSHOCK.EQ.1) GO TO 420
      DO 60 KK = 2, IS1
      KKM1 = KK - 1
      MA(KKM1) = IK
      TE = ADT(KK, KKM1)
C     PRINT 850,DTMAX, TE
      IF(X(1,KK) .LT. 0.9E0*X(1,IFF)) GO TO 60
      IF(X(1,KK) .LT.0.99E0*X(1,IFF)) GO TO 60
      DTMAX=3.0E0*DT
      IF(TE .LT. DTMAX) GO TO 60
      NN = (X(1, KK) - X(1, KKM1))/(((C(1, KK) + C(1, KKM1))*DT)*2
     1.
      IF(NN .LE. 1) GO TO 60
      IK = IK + NN - 1
   60 CONTINUE
      MA(IS1) = IK
      IS1P = IS1 + IK
      IS2P1 = IS2 + 1
      GO TO 421
 420  IS2P1=2
 421  CONTINUE
      DO 65 KK = IS2P1, INT1
      KKM1 = KK - 1
      MA(KKM1) = IK
      TE = ADT(KK, KKM1)
      DTMAX=1.8E0*DT
      IF(TE .LT. DTMAX) GO TO 65
      NN = (X(1, KK) - X(1, KKM1))/(((C(1, KK) + C(1, KKM1))*DT)*2
     1.
      IF(NN .LE. 1) GO TO 65
```

```
      IK = IK + NN - 1
   65 CONTINUE
      MA(INT1) = IK
      INT1P = INT1 + IK
      INT2P1 = INT2 + 1
      DO 70 KK = INT2P1, IS3
      KKM1 = KK - 1
      MA(KKM1) = IK
      TE = ADT(KK, KKM1)
      DTMAX=3.0E0*DT
      IF(TE .LT. DTMAX) GO TO 70
      NN = (X(1, KK) - X(1, KKM1))/((C(1, KK) + C(1, KKM1))*DT)*2
     1.
  302 FORMAT(1H ,"KK=",I4,"NN=",I4,"TE=",E15.8,"DT=",E15.8)
      IF(NN .LE. 1) GO TO 70
      IK = IK + NN - 1
   70 CONTINUE
      MA(IS3) = IK
      IS3P = IS3 + IK
      IS4P1 = IS4 + 1
      DO 75 KK = IS4P1, IFF
      KKM1 = KK - 1
      MA(KKM1) = IK
      IF(IFF .GT. 0.) GO TO 75
      TE = ADT(KK, KKM1)
      IF(TE .LT. DTMAX) GO TO 75
      NN = (X(1, KK) - X(1, KKM1))/((C(1, KK) + C(1, KKM1))*DT)*2
     1.
      IF(NN .LE. 1) GO TO 75
      IK = IK + NN - 1
   75 CONTINUE
      MA(IFF) = IK
      IFFP = IFF + IK
  303 FORMAT(1H ,"IFF=",I4,"IFFP=",I4,"IK=",I4)
      IF(MA(IFF) .GE. 1) GO TO 76
      GO TO 77
   76 X(1, IFFP) = X(1, IFF)
      P(1, IFFP) = P(1, IFF)
      U(1, IFFP) = U(1, IFF)
      E(1, IFFP) = E(1, IFF)
      RH(1, IFFP) = RH(1, IFF)
      C(1, IFFP) = C(1, IFF)
      Q(1, IFFP) = Q(1, IFF)
   77 CONTINUE
      K = IFFP
      DO 80 KK = 2,IFF
      IN = IFF - KK + 1
      NPT = MA(IN + 1) - MA(IN)
      K = K - NPT - 1
      X(1, K) = X(1, IN)
```

```
      P(1, K) = P(1, IN)
      U(1, K) = U(1, IN)
      E(1, K) = E(1, IN)
      RH(1, K) = RH(1, IN)
      C(1, K) = C(1, IN)
      Q(1, K) = Q(1, IN)
   80 CONTINUE
C     CALCULATE PROPERTIES AT ADDED PTS.
      KCHCK = 1
        IF(KSHOCK.EQ.1) KCHCK=2
      K = 1
   79 GO TO (81, 82, 83, 84),KCHCK
   81 M1 = 2
      M2 = IS1 -1
      GO TO 90
   82 M1 = IS2 + 1
       IF(KSHOCK.EQ.1) M1=2
      M2 = INT1 -1
      IF(M1 .EQ. INT1) GO TO 92
      GO TO 90
   83 M1 = INT2 + 1
      M2 = IS3 -1
      IF(M1 .EQ. IS3) GO TO 93
      GO TO 90
   84 M1 = IS4 + 1
      M2 = IFF -1
      IF(M1 .EQ. IFF) GO TO 98
   90 CONTINUE
      DO 86 KK = M1, M2
      NPT2 = MA(KK + 1) - MA(KK)
      NPT1 = MA(KK) - MA(KK - 1)
      K = K + NPT1 + 1
      IF(NPT1 .EQ. 0) GO TO 86
      IF(NPT1 .LT. 0) PRINT 814
      XP1 = NPT1 + 1
      DX = X(1, K) - X(1, K - NPT1 - 1)
      DY1 = -DX/XP1
      DX1 = DX
      DX2 = X(1, K + NPT2 + 1) - X(1, K)
      DY = DY1
      DO 85 IA = 1, NPT1
      P2 = P(1, K)
      U2 = U(1, K)
      E2 = E(1, K)
      RH2 = RH(1, K)
      Q2 = Q(1, K)
      C2 = C(1, K)
      DP1 = P(1, K) - P(1, K - NPT1 - 1)
      DU1 = U(1, K) - U(1, K - NPT1 - 1)
      DE1 = E(1, K) - E(1, K - NPT1 - 1)
```

```
      DRH1 = RH(1, K) - RH(1, K - NPT1 - 1)
      DQ1 = Q(1, K) - Q(1, K - NPT1 - 1)
      DC1 = C(1, K) - C(1, K - NPT1 - 1)
      DP2 = P(1, K + NPT2 + 1) - P(1, K)
      DU2 = U(1, K + NPT2 + 1) - U(1, K)
      DE2 = E(1, K + NPT2 + 1) - E(1, K)
      DRH2 = RH(1, K + NPT2 + 1) - RH(1, K)
      DQ2 = Q(1, K + NPT2 + 1) - Q(1, K)
      DC2 = C(1, K + NPT2 + 1) - C(1, K)
      P(1, K - IA) = QINP(P2, DP1, DP2, DY)
      U(1, K - IA) = QINP(U2, DU1, DU2, DY)
      E(1, K - IA) = QINP(E2, DE1, DE2, DY)
      RH(1, K - IA) = QINP(RH2, DRH1, DRH2, DY)
      Q(1, K - IA) = QINP(Q2, DQ1, DQ2, DY)
      C(1, K - IA) = QINP(C2, DC1, DC2, DY)
      X(1, K - IA) = X(1, K) + DY
      XIA = IA + 1
      DY = DY1*XIA
C     PRINT 800, I,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J),L
   85 CONTINUE
      J = K - IA
   86 CONTINUE
      GO TO (91, 92, 93, 94), KCHCK
   91 M3 = IS1
      GO TO 96
   92 M3 = INT1
      GO TO 96
   93 M3 = IS3
      GO TO 96
   94 GO TO 98
   96 CONTINUE
      NPT2 = MA(M3) - MA(M3 - 1)
      K = K + NPT2 + 1
      IF(NPT2 .LE. 0) GO TO 97
      XP2 = NPT2 + 1
      DX = X(1, K) - X(1, K - NPT2 - 1)
      DY1 =-DX/XP2
      DY = DY1
      DO 95 IA = 1, NPT2
      P2 = P(1, K)
      U2 = U(1, K)
      E2 = E(1, K)
      RH2 = RH(1, K)
      Q2 = Q(1, K)
      C2 = C(1, K)
      DP2 = P(1, K) - P(1, K - NPT2 - 1)
      DU2 = U(1, K) - U(1, K - NPT2 - 1)
      DE2 = E(1, K) - E(1, K - NPT2 - 1)
      DRH2 = RH(1, K) - RH(1, K - NPT2 - 1)
      DQ2 = Q(1, K) - Q(1, K - NPT2 - 1)
```

```
      DC2 = C(1, K) - C(1, K - NPT2 - 1)
      J = K - IA
      P(1, J                    ) = XLINP(P2, DP2, DX, DY)
      U(1, J                    ) = XLINP(U2, DU2, DX, DY)
      E(1, J                    ) = XLINP(E2, DE2, DX, DY)
      Q(1, J                    ) = XLINP(Q2, DQ2, DX, DY)
      C(1, J                    ) = XLINP(C2, DC2, DX, DY)
      RH(1, J                    ) = XLINP(RH2, DRH2, DX, DY)
      X(1, J                    ) = X(1, K) + DY
      XIA = IA + 1
      DY = DY1*XIA
C     PRINT 800, I,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J),L
   95 CONTINUE
   97 CONTINUE
      K = K + 1
      KCHCK = KCHCK + 1
      IF(KCHCK .LE. 4) GO TO 79
C   -   COMPLETE ADD PTS.
   98 CONTINUE
      DO 101 J = 2, IFF
      IF(MA(J) .EQ. MA(J-1)) GO TO 101
      N = MA(J) - MA(J-1)
      JM1 = J - 1
  101 CONTINUE
      IS1 = IS1P
      IS2 = IS1P + 1
      IS3 = IS3P
      IS4 = IS3P + 1
      INT1 = INT1P
      INT2 = INT1P + 1
      IFF = IFFP
C     PRINT 1000, IS1, IS2, INT1, INT2, IS3, IS4, IFF
C     DO 99 J = 1, IFF
C  99 PRINT 800, I,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J),L
  100 CONTINUE
C     DO 250 J = 1, IFF
C 250 PRINT 800, I,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J),L
      XIS1 = X(1, IS1) + UU1*DT1
      XIS3 = X(1, IS3) + UU2*DT1
      XIS2 = XIS1
      XIS4 = XIS3
      XINT = X(1, INT1) + U(1, INT1)*DT1
C   -   IK = NO PTS ELIMINATED.
C   -   ME(IK) = PT OF ORIGINAL NO BEING ELIMINATED
      IK = 0
       IF(KSHOCK.EQ.1) GO TO 116
      IS1M1 = IS1 - 1
      IF ( UU1 .GT. 0. ) GO TO 107
      DO 105 KK = 2, IS1M1
      XD = XIS1 - (X(1, KK) + U(1, KK)*DT1)
```

```
      IF(XD .GT. 0.) GO TO 105
      IK = IK + 1
      ME(IK) = KK
  105 CONTINUE
      IS1P = IS1 - IK
      GO TO 116
  107 IS1P = IS1 - IK
  110 CONTINUE
      IS2P1 = IS2 + 1
      INT1M1 = INT1 - 1
      IF ( UU1 .LT. 0. ) GO TO 116
      DO 115 KK = IS2P1, INT1M1
C     XD = XIS1 - (X(1,KK) + U(1,KK) * DT1 )
      XD = XIS1 - (X(1,KK) + U(1,KK) * DT1 )  -0.00100
      IF(XD .LT. 0.) GO TO 115
      IK = IK + 1
      ME(IK) = KK
  115 CONTINUE
  116 CONTINUE
      INT1P = INT1 - IK
  120 CONTINUE
      INT2P1 = INT2 + 1
      IS3M1 = IS3 - 1
      IF ( UU2 .GT. 0. ) GO TO 127
      DO 125 KK = INT2P1, IS3M1
C     XD = XIS3 - ( X(1,KK) + U(1,KK) * DT1)
      XD = XIS3 - ( X(1,KK) + U(1,KK) * DT1) -0.00100
      IF(XD .GT. 0.) GO TO 125
      IK = IK + 1
      ME(IK) = KK
  125 CONTINUE
      IS3P = IS3 - IK
      GO TO 136
  127 IS3P = IS3 -IK
  130 CONTINUE
      IS4P1 = IS4 + 1
      IFFM1 = IFF - 1
      IF ( UU2 .LT. 0. ) GO TO 136
      DO 135 KK = IS4P1 , IFFM1
      XD = XIS4 - (X(1,KK) + U(1, KK) * DT1)
      IF(XD .LT. 0.) GO TO 135
      IK = IK + 1
      ME(IK) = KK
  135 CONTINUE
  136   CONTINUE
      IFFP = IFF - IK
  140 CONTINUE
      IF ( IK .EQ. 0) GO TO 161
      K = 1
      JK = 1
```

```
      DO 150 KK = 1, IFF
      IF ( ME(JK) .EQ. KK) GO TO 145
      P(1, K) = P(1, KK)
      U(1, K) = U(1, KK)
      RH(1, K) = RH(1, KK)
      E(1, K) = E(1, KK)
      C(1, K) = C(1,KK)
      Q(1, K) = Q(1, KK)
      X(1, K) = X(1, KK)
      K = K + 1
      GO TO 150
  145 CONTINUE
      IF(JK .EQ. IK) GO TO 150
      JK = JK + 1
  150 CONTINUE
      IF(IK .EQ. 0) GO TO 161
  146 CONTINUE
      DO 160 J = 1, IK
  160 CONTINUE
  161 IS1 = IS1P
      IS2 = IS1P + 1
      IS3 = IS3P
      IS4 = IS3P + 1
      INT1 = INT1P
      INT2 = INT1P + 1
      IFF = IFFP
C     DO 251 J = 1 , IFF
C 251 PRINT 800, I,J,X(1,J),U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J),L
      RETURN
      END



      SUBROUTINE RASHOK(L, IN, MSR)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),    E(2,1000),P(2,1000)
      COMMON/INIT/PI1,UI1,RHI1,EI1,CI1,QI1,PI2,UI2,RHI2,EI2,CI2,Q
     1I2
      COMMON / SHK4B / U4B,C4B,RH4B,E4B,P4B,X4B,Q4B     ,UUU
      COMMON /SHK4A / U4A,C4A,RH4A,E4A,P4A,X4A,Q4A
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/RAWAV / XR(30),UR(30),CR(30),RHR(30),ER(30),PR(30),R
     1HP(30)
      COMMON /C1AND2/ PF1,UF1,RHF1,EF1,CF1,QF1,PF2,UF2,RHF2,EF2,C
     1F2,QF2,  XF,XS
      COMMON/SINPT/ PI, UI, RHI, UF, XZ, PF, IR1, IR2
C      IN   -  = OF INTERVALS DIV. FOR PRESSURE.,   MSR = 1   -
C       RIGHT S
C     AND LEFT RAWAVE,   MSR = 2   -  LEFT SHOCK AND RIGHT
C       RAWAVE.
```

```
      NIT=100
      TOL = 0.0005
      MAX=100
      IR1 = 1
      IR2 = IN + 1
      K = 1
      PI = PI1
      UI = UI1
      P4A = PI2
      RH1 = RHI1
      RH4A = RHI2
      E4A = EI2
      C4A = CI2
      U4A = UI2
      MS = 1
      MR = 1
      LA=1
      LS=2
      IF (MSR .EQ. 1) GO TO 10
      MS = 2
      MR = 2
      LA=2
      LS=1
   10 CONTINUE
      PF = (PI + P4A)/2.
      P4B = PF
      CALL RAWAVE (LA,IN,MR)
      CALL SHOKEQ(LS,MS,1)
   15 CONTINUE
      U4B = (U4B + UF)/2.
      CALL SHOKEQ (LS,MS,2)
      PF = P4B
      CALL RAWAVE (LA,IN,MR)
      IF (ABS((U4B-UF)/UF) .LT. TOL) GO TO 30
      IF (K .LT. NIT) GO TO  20
      GO TO 30
   20 CONTINUE
      GO TO 15
   30 CONTINUE
      INF = IN + 1
      DO 40 J = 1, INF
      X(1,J) = XR(J)
      P(1,J) = PR(J)
      U(1,J) = UR(J)
      E(1,J) = ER(J)
      RH(1,J) = RHR(J)
      C(1,J) = CR(J)
      Q(1,J) = 0.
   40 CONTINUE
      INT1 = INF + 1
```

```
      INT2 = INT1 + 1
      X(1,INT1) = XZ + UF*DT
      P(1,INT1) = PF
      RH(1,INT1) = RHR(INF)
      E(1,INT1) = ER(INF)
      U(1,INT1) = UF
      C(1,INT1) = CR(INF)
      Q(1,INT1) = 0.
      X(1,INT2) = X(1,INT1)
      P(1,INT2) = PF
      RH(1,INT2) = RH4B
      E(1,INT2) = E4B
      U(1,INT2) = UF
      C(1,INT2) = C4B
      Q(1,INT2) = 0.
      IS3 = INT2 + 1
      IS4 = IS3 + 1
      X(1,IS3) = XZ + UUU*DT
      P(1,IS3) = PF
      RH(1,IS3) = RH4B
      U(1,IS3) = UF
      E(1,IS3) = E4B
      Q(1,IS3) = 0.
      C(1,IS3) = C4B
      X(1,IS4) = X(1,IS3)
      P(1,IS4) = P4A
      RH(1,IS4) = RH4A
      E(1,IS4) = E4A
      C(1,IS4) = C4A
      U(1,IS4) = U4A
      Q(1,IS4) = 0.
      IX = 1
      XF = X(1, INT1)
      XS = X(1, IS3)
      PF1 = P(1, INT1)
      UF1 = U(1, INT1)
      RHF1 = RH(1, INT1)
      EF1 = E(1, INT1)
      CF1 = C(1, INT1)
      QF1 = Q(1, INT1)
      PF2 = P(1, INT2)
      UF2 = U(1, INT2)
      RHF2 = RH(1, INT2)
      EF2 = E(1, INT2)
      CF2 = C(1, INT2)
      QF2 = Q(1, INT2)
  914 FORMAT(1H0,I5,5X,6(1PE15.8))
  922 FORMAT(1H1,"THESE POINTS DEFINE THE INITIAL SINGULARITY AND
     1UT THE INTERFACE"/
     2                1H0,"POINT NO. ",7X,"X",14X,"U",14X,"C",14X,"
```

```
     3P",13X,"RH",14X,"E"/ 1H ,"RAREFACTION IN REGION ONE")
 924 FORMAT(1H ,"INTERFACE")
 925 FORMAT(1H ,"SHOCK IN REGION TWO")
 926 FORMAT(1HO,"SHOCK VELOCITY ",1PE15.8)
     PRINT 922
     A=1.0
     DO 50 J=1,IS4
     IF(J.EQ.INT1)PRINT 924
     IF(J.EQ.IS3)PRINT 925
     PRINT 914,J,A,U(1,J),C(1,J),P(1,J),RH(1,J),E(1,J)
  50 CONTINUE
     PRINT 926,UUU
     RETURN
     END



     SUBROUTINE RAWAVE(L, IN, MR)
C    -   COMPUTE RAREFACTION WAVE.
C    -   RHI, PI, UI; INITIAL PROPERTIES.
C    -   PF; PRESSURE AT CONTACT LINE.
C    -   IN; NO. OF INTERVALS TO BE DIVIDED FOR PRESSURE.
C    -   MR = 1, LEFT RUNNING WAVE; MR = 2, RIGHT RUNNING WAVE.
     COMMON/RAWAV / X(30),U(30),C(30),RH(30),E(30),P(30),RHP(30)
     COMMON/SINPT/ PI, UI, RHI, UF, XZ, PF, IR1, IR2
     COMMON/TIMUU/ DT, UU1, UU2, I, XMU
 100 FORMAT("  NOT CONVERGING IN RAWAVE ON N =", I2)
 900 FORMAT(12X, "X", 16X, "U", 16X, "C", 17X, "P", 17X, "RH",17
    1X,"E")
1000 FORMAT(4X, 6(E15.8, 2X), I2)
 101 FORMAT(1H ,"ROUTINE RAWAVE    L=",I3,"  IN=",I3,"  MR=",I3)
     TOL = 0.0001
     MAX=100
     SIGN = -1.
     IF(MR .EQ. 2) SIGN =  1.
C    -   DEFINE PROPERTIES.
     P(1) = PI
     U(1) = UI
     RH(1) = RHI
     E(1) = EQSTEQ(L, RHI, PI)
     C(1) = EQSTCQ(L, E(1), RHI, PI)
     X(1) = XZ + (U(1) + SIGN*C(1))*DT
     XIN = IN
     DP = (PF - PI)/XIN
     INF = IN + 1
     DO 5 N = 2, INF
   5 P(N) = P(N-1) + DP
     XINF = INF
     DO 25 N = 2, INF
C    -   ASSUME VALUES FOR VARIABLES.
```

```
      K = 1
       RH(N)=RH(N-1) * (1.+1./XINF*SIGN)
      C(N) = C(N-1)
      RHC = RH(N-1)*C(N-1)
C   -   CALCULATION OF PROPERTIES.
   10 CONTINUE
      RHC = (RH(N-1)*C(N-1) + RH(N)*C(N))/2.
      PRH = (P(N-1)/(RH(N-1)**2) + P(N)/(RH(N)**2))/2.
      U(N) = SIGN*(P(N) - P(N-1))/RHC + U(N-1)
      E (N) = PRH*(RH(N) - RH(N-1)) + E(N-1)
      RHP(N) = EQSTRQ(L, E (N), P(N))
      C(N) = EQSTCQ(L, E (N), RHP(N), P(N))
 1011 FORMAT(1H ,"RHP(N)=",E15.8,"  C(N)=",E15.8)
  118 FORMAT(1HO,"RHP=",E13.6,"E=",E13.6,"U=",E13.6,             "C
     1=",E13.6,"RHC=",E13.6,"PRH=",E13.6)
      IF( ABS((RHP(N) - RH(N))/RH(N)) .GT. TOL) GO TO 20
      GO TO 22
   20 CONTINUE
      IF(K .LT. MAX) GO TO 21
      PRINT 100, N
      GO TO 22
C     GO TO 30
   21 CONTINUE
      IF(K .GE. 15) GO TO 211
      RH(N)=( RH(N)+RHP(N)) / 2.
      GO TO 212
  211 FF=UP-U(N)
      FFRH=0.5E0*SIGN*C(N)*(P(N)-P(N-1))/RHC**2
      RH(N)=RH(N)-FF / FFRH
      IF(UP .LT. 0.1E-6) GO TO 2111
      FF=FF/UP
 2111 CONTINUE
      TAL=TOL*1.E-2
      IF( ABS(FF) .LT. TAL) GO TO 22
 1010 FORMAT(1H ,"RH(N)=",E15.8,"FF=",E15.8,"FFRH=",E15.8)
  212 CONTINUE
      C(N)=EQSTCQ(L,E(N),RH(N),P(N))
      UP=U(N)
      K = K + 1
      GO TO 10
   22 CONTINUE
      X(N) = XZ + (U(N) + SIGN*C(N))*DT
C     PRINT 900
   25 CONTINUE
      PF = P(INF)
      UF = U(INF)
  110 FORMAT(1H ,"   PF =  ",E15.8,"   UF =  ",E15.8)
   30 RETURN
      END
```

```
      SUBROUTINE REFLCK(IS1, XUU)
C   -   SUBROUTINE TO CHECK REFLECTION OF SECOND SHOCK.
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON/TIMUU/ DT, UU1, UU2, I,XMU
      XUU = X(1, IS1) + UU1*DT
      RETURN
      END


      SUBROUTINE REFLSK(L)
C   -   SUBROUTINE FOR SHOCK REFLECT AT THE CENTER.
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/NOCON/ IS1, IS2, IS3, IS4, INT1, INT2, IMAX
      COMMON /SHK4A / U4A,C4A,RH4A,E4A,P4A,X4A,Q4A
      COMMON / SHK4B / U4B,C4B,RH4B,E4B,P4B,X4B,Q4B,UUU
      COMMON/SHKI/ EP
  800 FORMAT("  SHOK BEING REFLECTED AT THE CENTER")
  801 FORMAT("  MACH NO FOR THE REFLECTED SHOCK =", E15.8, /, "
     1VELOCITY OF THE SECOND SHOCK =", E15.8)
  802 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X), I2)
  803 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X), " REFLSHK")
      L = 1
      XM = -UU1/C(1, IS2)
      UU1 = -UU1
      X(2, IS2) = X(1, IS2)
      P(2, IS2) = P(1, IS2)
      U(2, IS2) = U(1, IS2)
      RH(2, IS2) = RH(1, IS2)
      E(2, IS2) = E(1, IS2)
      C(2, IS2) = C(1, IS2)
      Q(2, IS2) = Q(1, IS2)
      U4A = U(2, IS2)
      C4A = C(2, IS2)
      RH4A = RH(2, IS2)
      E4A = E(2, IS2)
      P4A = P(2, IS2)
      CALL SHOKEQ(L, 1, 3)
      X(2, IS1) = X(2, IS2)
      P(2, IS1) = P4B
      U(2, IS1) = U4B
      RH(2, IS1) = RH4B
      E(2, IS1) = E4B
      C(2, IS1) = C4B
      Q(2, IS1) = Q(2, IS2)
      IS1M1 = IS1 - 1
      DO 10 K = 1, IS1M1
```

```
      X(2, K) = X(1, K)
      P(2, K) = P(2, IS1)
      RH(2, K) = RH(2, IS1)
      E(2, K) = E(2, IS1)
      C(2, K) = C(2, IS1)
      Q(2, K) = Q(2, IS1)
   10 CONTINUE
      XIS1M1 = IS1 - 1
      DU = U(2, IS1)/XIS1M1
      U(2, 1) = 0.
      DO 15 K = 2, IS1M1
   15 U(2, K) = U(2, K-1) + DU
      DO 21 J = IS1,IS2
   21 PRINT 803,I,J,X(2,J),U(2,J),C(2,J),P(2,J),RH(2,J),E(2,J)
      RETURN
      END


      SUBROUTINE SHFPT(L, I1, I2, I3, I4, I5, I6, UU)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON /SHK4A / U4A,C4A,RH4A,E4A,P4A,X4A,Q4A
      COMMON / SHK4B / U4B,C4B,RH4B,E4B,P4B,X4B,Q4B      ,UUU
      COMMON/SHKI/ EP
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/GNPSBB/ XB
      COMMON / REFE / TREF,T
C 800 FORMAT("  XB =", E15.8, " UB =", E15.8, " CB =", E15.8,"
C        X4B =",
C    1  E15.8)
C 801 FORMAT("  CALCULATION OF ITERATION IN SHOCK FRONT POINT")
C 802 FORMAT(" P4B =", E15.8, " U4B =", E15.8, " E4B =", E15.8,
C    1  " RH4B =", E15.8, " C4B =", E15.8, " X4B =", E15.8)
C 805 FORMAT("  SHOCK DOES NOT EXIST; INITIATE SHOK")

C 810 FORMAT("  MS BEING CHANGED, MS =", I2)
 820  FORMAT(1H ,4X,"SHOCK VELOCITY FOR  THE LEFT SHOCK UU1 =",E1
     15.8)
 821  FORMAT(1H , 4X,"SHOCK VELOCITY FOR THE RIGHT SHOCK UU2 =",
     1E15.8)
 850   FORMAT(1H , 6(I4,2X))
  860  FORMAT (1H , 6(E15.8))
  900 FORMAT(1H , I2, 1X, I4, 3X, 6(E15.8, 2X), I2, 2X, " SF")
 1000 FORMAT("  SHOCK FRONT POINT NOT CONVERGING.")
      QINP( V2, DV1, DV2, DY) = V2 + (DV2*DX1**2+DV1*DX2**2)* DY
     1/ (       DX1*DX2*(DX1 + DX2)) + (-DV1*DX2 + DV2*DX1)*(  DY
     2 )**2/(DX1*DX2    *(DX1 + DX2))
C   -    MS = 1, RIGHT RUNNING SHOCK; MS = 2, LEFT RUNNING SHOCK.
C     IF ( UU .GE. 0.) MS = 1
```

```
C       IF(UU .LT. 0.   ) MS = 2
C       IF(L .EQ. 2) GO TO 3
C       IF(T .GT.  TREF .AND. L .EQ. 1) GO TO 3
C       IF(UU .LE. 0.) GO TO 3
C       MS = 2
C       SIGN = -1.
C       GO TO 50
C     3 CONTINUE
C       MK = MS
        MS = 2
        IF(U(1, I3) .LT. UU) MS = 1
        IF(U(1, I3) .GT. UU) MS = 2
        TOL = 0.0005
        TOL1 = 1.E-10
C       NIT = 20
              NIT=30
        EP = 0.01
        ICHCK = 1
        K = 1
C     -   DEFINE PROPERTIES.
      5 CONTINUE
        IF(MS .EQ. 1) SIGN = 1.
        IF(MS .EQ. 2) SIGN = -1.
        IF(MS .EQ. 2) GO TO 7
        X2 = X(1, I2)
        U2 = U(1, I2)
        C2 = C(1, I2)
        E2 = E(1, I2)
        P2 = P(1, I2)
        RH2 = RH(1, I2)
        Q2 = Q(1, I2)
        DX1 = X(1, I2) - X(1, I1)
        DU1 = U(1, I2) - U(1, I1)
        DC1 = C(1, I2) - C(1, I1)
        DE1 = E(1, I2) - E(1, I1)
        DRH1 = RH(1, I2) - RH(1, I1)
        DP1 = P(1, I2) - P(1, I1)
        DQ1 = Q(1, I2) - Q(1, I1)
        DX2 = X(1, I3) - X(1, I2)
        DU2 = U(1, I3) - U(1, I2)
        DC2 = C(1,I3) - C(1, I2)
        DE2 = E(1, I3) - E(1, I2)
        DRH2 = RH(1, I3) - RH(1, I2)
        DP2 = P(1, I3) - P(1, I2)
        DQ2 = Q(1, I3) - Q(1, I2)
        GO TO 8
      7 CONTINUE
        X2 = X(1, I5)
        U2 = U(1, I5)
        C2 = C(1, I5)
```

```
      E2 = E(1, I5)
      P2 = P(1, I5)
      RH2 = RH(1, I5)
      Q2 = Q(1, I5)
      DX1 = X(1, I5) - X(1, I4)
      DU1 = U(1, I5) - U(1, I4)
      DC1 = C(1, I5) - C(1, I4)
      DE1 = E(1, I5) - E(1, I4)
      DRH1 = RH(1, I5) - RH(1, I4)
      DP1 = P(1, I5) - P(1, I4)
      DQ1 = Q(1, I5) - Q(1, I4)
      DX2 = X(1, I6) - X(1, I5)
      DU2 = U(1, I6) - U(1, I5)
      DC2 = C(1, I6) - C(1, I5)
      DE2 = E(1, I6) - E(1, I5)
      DRH2 = RH(1,I6) - RH(1, I5)
      DP2 = P(1, I6) - P(1, I5)
      DQ2 = Q(1, I6) - Q(1,I5)
C   -   COMPUTE X4B
    8 CONTINUE
      X4B = X(1, I3) + UU*DT
      UB = U2
      CB = C2
      XB = X2
C   -   COMPUTE XB BY ITERATION.
    9 CONTINUE
      DXB = XB - X2
      XBP = X4B - (UB + SIGN*CB)*DT
      UB = QINP(U2, DU1, DU2, DXB)
      CB = QINP(C2, DC1, DC2, DXB)
      IF( ABS((XBP - XB)/XBP) .LT. TOL) GO TO 10
      XB = (XB + XBP)/2.
C     PRINT 800, XB, UB, CB, X4B
      GO TO 9
   10 CONTINUE
C     IF(MS .EQ. 2) GO TO 20
C     IF(XB .GE. X(1, I3)) GO TO 30
C     GO TO 22
C  20 IF(XB .LE. X(1, I3)) GO TO 30
C  22 CONTINUE
C     PRINT 801
      U4B = U2
   23 CONTINUE
      IF(MS .EQ. 1) GO TO 29
      CALL GNPSHB (L, I4,I5,I6,X4B,MS )
      GO TO 31
   29 CALL GNPSHB(L, I1, I2, I3, X4B, MS)
   31 U4B1 = U4B
   24 IF(MS .EQ. 1) GO TO 25
      CALL GNPSHA(L, I1, I2, I3, X4B)
```

```
      GO TO 26
   25 CALL GNPSHA(L, I4, I5, I6, X4B)
   26 CALL SHOKEQ(L, MS, 1)
      IF((K-(K/2)*2) .EQ. 0) GO TO 265
      UUUA=UUU
      U4BA=U4B
      GO TO 268
  265 UUUB=UUU
      U4BB=U4B
  268 IF(K .LT.10) GO TO 269
      IF(ABS(UUUA-UUUB) .LT.0.1E-07) GO TO 269
      BM=(U4BA-U4BB) / (UUUA-UUUB)
      U4B=U4BB+ 0.5*(UUUA-UUUB)*BM
  269 IF( ABS(U4B1) .LT. TOL1) GO TO 27
      IF( ABS((U4B1 - U4B)/U4B1) .LT. TOL) GO TO 60
   27 CONTINUE
      IF(K .GE. NIT) GO TO 28
      U4B = (U4B1 + U4B)/2.
      K = K + 1
C     PRINT 802, P4B, U4B, E4B, RH4B, C4B, X4B
      GO TO 23
   28 PRINT 1000
      GO TO 50
C  30 CONTINUE
C     PRINT 805
      IF(L .EQ. 2) GO TO 50
      IF ( T .LT. TREF ) MS = 2
      IF ( T .GE. TREF ) MS = 1
      IF(MS .EQ. 1) SIGN = 1.
      IF(MS .EQ. 2) SIGN = -1.
      GO TO 50
C     IF(L .EQ. 2) GO TO 50
C    -  SHOCK DOES NOT EXIST.
C     IF(ICHCK .EQ. 2) GO TO 50
C     IF(MS .EQ. 1) IS = I3
C     IF(MS .EQ. 2) IS = I4
C  40 CONTINUE
C     ICHCK = 2
C     P4A = P(1, IS)
C     U4A = U(1, IS)
C     E4A = E(1, IS)
C     RH4A = RH(1, IS)
C     C4A = C(1, IS)
C     Q4A = Q(1, IS)
C     IF(MK .EQ. 1) MS = 2
C     IF ( MK .EQ. 2) MS = 1
C     PRINT 810, MS
C     CALL SHOKIN(L, MS, 1, IS)
C     GO TO 5
   50 CONTINUE
```

```
      IF(L .EQ. 2) GO TO 47
      UU1 = U(1,I3)+ SIGN*C(1,I3)
      GO TO 48
   47 UU2 = U(1,I3)+ SIGN*C(1, I3)
   48 X4B = X(1, I3) + (U(1, I3) + SIGN*C(1, I3))*DT
      IF ( L.EQ. 2) GO TO 55
      IF ( MS .EQ. 1) GO TO 55
      CALL GNPSHA(L, I1, I2, I3, X4B)
      GO TO 58
   55 CONTINUE
      CALL GNPSHA(L, I4, I5, I6, X4B)
   58 CONTINUE
      P4B = P4A
      U4B = U4A
      E4B = E4A
      RH4B = RH4A
      C4B = C4A
      Q4B = Q4A
   60 CONTINUE
         EP = (P4B - P4A)/P4A
      IF(L .EQ. 1 .AND. MS .EQ. 2) GO TO 70
      X(2, I3) = X4B
      Q(2, I3) = Q4B
      P(2, I3) = P4B
      E(2, I3) = E4B
      RH(2, I3) = RH4B
      C ( 2,I3) = C4B
      U(2, I3) = U4B
   67 X(2, I4) = X4B
      Q(2, I4) = Q4A
      P(2, I4) = P4A
      E(2, I4) = E4A
      RH(2, I4) = RH4A
      C(2, I4) = C4A
      U(2, I4) = U4A
      GO TO  75
   70 CONTINUE
      X(2, I3) = X4B
      Q(2, I3) = Q4A
      P(2, I3) = P4A
      E(2, I3) = E4A
      RH(2, I3) = RH4A
      C(2, I3) = C4A
      U(2, I3) = U4A
      X(2, I4) = X4B
      Q(2, I4) = Q4A
      P(2, I4) = P4B
      E(2, I4) = E4B
      RH(2, I4) = RH4B
      C(2, I4) = C4B
```

```
      U(2, I4) = U4B
   75 CONTINUE
      IF ( ABS(EP) .GT. 0. ) GOTO 79
         IF(MS .EQ. 2) GO TO 78
         X(2, I3) = X(2, I4)
         Q(2, I3) = Q(2, I4)
         P(2, I3) = P(2, I4)
         E(2, I3) = E(2, I4)
         RH(2, I3) = RH(2, I4)
         C(2, I3) = C(2, I4)
          U(2, I3) = U(2, I4)
         GO TO 79
   78    CONTINUE
         X(2, I4) = X(2, I3)
         Q(2, I4) = Q(2, I3)
         P(2, I4) = P(2, I3)
         E(2, I4) = E(2, I3)
         RH(2, I4) = RH(2, I3)
         C(2, I4) = C(2, I3)
         U(2, I4) = U(2, I3)
 79   CONTINUE
      IF ( L .EQ. 1 ) GO TO 80
      GO TO 90
 80   CONTINUE
   90 CONTINUE
      IF ( L  .EQ. 1 ) UU = UU1
      IF ( L .EQ. 2 ) UU = UU2
      RETURN
      END


      SUBROUTINE SHOKIN (L,MS,MQ,IS)
      COMMON/GAIN/ Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,1
     1000),     E(2,1000),P(2,1000)
      COMMON/NOCON/ IS1, IS2, IS3, IS4, INT1, INT2, IMAX
      COMMON / SHK4B / U4B,C4B,RH4B,E4B,P4B,X4B,Q4B     ,UUU
      COMMON /SHK4A / U4A,C4A,RH4A,E4A,P4A,X4A,Q4A
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/SHKI/ EP
C       IS    -   POSITION TO INITIATE SHOCK
C       MS = 1    -  RIGHT SHOCK,    MS = 2    -  LEFT SHOCK
C       MQ = "    -  GIVEN P2,    MQ = 2    -  GIVEN U2
 1000 FORMAT(1H ,I2,1X,I4,3X,6(E15.8,2X),I2,2X,"INSK")
 1001 FORMAT("  INITIATED SHOCK SPEED UU2 =", E15.8)
 1002 FORMAT("  INITIATED SHOCK SPEED UU1 =", E15.8)
      TOLSK = 1.E-06
      TOLSK=0.0001
      IF ( MS .EQ. 1) SIGN = 1.
      IF ( MS .EQ.  2 ) SIGN = -1.
```

```
      IS1 = IS
      IS2 = IS + 1
      U4A = U(1, IS)
      C4A = C(1, IS)
      RH4A = RH(1, IS)
      E4A = E(1, IS)
      P4A = P(1, IS)
      Q4A = Q(1, IS)
      IF ( ABS(EP) .LT. TOLSK ) GO TO 7
      P4B = P4A * EP + P4A
      CALL SHOKEQ (L,MS, 1)
      GO TO 8
    7 CONTINUE
      UU1 = U(1,IS) + SIGN * C(1,IS)
      CALL SHOKEQ ( L,MS, 3)
    8 CONTINUE
      IF(MS .EQ. 1) GO TO 10
      U(1,IS1) = U4A
      P(1,IS1) = P4A
      RH (1,IS1) = RH4A
      E(1,IS1) = E4A
      C(1,IS1) = C4A
      Q(1,IS1) = Q4A
      X(1,IS1) = X(1,IS)
      U(1,IS2) = U4B
      P(1,IS2) = P4B
      RH(1,IS2) = RH4B
      E(1,IS2) = E4B
      C(1,IS2) = C4B
      Q(1,IS2) = Q(1,IS)
      X(1,IS2) = X(1,IS)
      GO TO 20
   10 CONTINUE
   11 U(1,IS1) = U4B
      P(1,IS1) = P4B
      RH(1,IS1) = RH4B
      E(1,IS1) = E4B
      C(1,IS1) = C4B
      Q(1,IS1) = Q(1,IS)
      X(1,IS1) = X(1,IS)
      U(1,IS2) = U4A
      P(1,IS2) = P4A
      RH(1,IS2) = RH4A
      E(1,IS2) = E4A
      C(1,IS2) = C4A
      Q(1,IS2) = Q(1,IS)
      X(1,IS2) = X(1,IS)
   20 CONTINUE
      IF(L .EQ. 1) GO TO 21
      GO TO 22
```

```
 21    CONTINUE
   22 CONTINUE
      RETURN
      END


      SUBROUTINE SHOKEQ (L,MS,MQ)
      COMMON /SHK4A / U4A,C4A,RH4A,E4A,P4A,X4A,Q4A
      COMMON / SHK4B / U4B,C4B,RH4B,E4B,P4B,X4B,Q4B,UUU
      COMMON/TIMUU/ DT, UU1, UU2, I, XMU
      COMMON/SHKI/ EP
C 900 FORMAT (1H ,  6( E15.8))
C 901 FORMAT(1H ," ROUTINE SHOKEQ   L=",I3,"  MS=",I2,"  MQ=",I4)
C      MS = 1    -  RIGHT SHOCK
C      MS = 2     -  LEFT SHOCK
C   -   MQ = 1, GIVEN P2;  MQ = 2, GIVEN U2;  MQ = 3, GIVEN UU.
C      MQ = 1     -  GIVEN P2, MQ =2      -  GIVEN U2
C     PRINT 910
C 910 FORMAT (1H ,   "  U2P   UUP   E2P    RH2P    C2P       P2")
  906  FORMAT (1H , " *** WARNING ****  RH2 IS LESS THAN RH1" )
 1000 FORMAT (1H ,"SHOCKEQ NOT CONVERGING")
C     PRINT 901,L,MS,MQ
 1313 FORMAT(6X,"RH1= ",E15.8,6X,"UU= ",E15.8,6X,"U1= ",E15.8)
      XS = 1.
      IF ( MS .EQ. 2) XS = -1.
      TOL = 0.0005
      TOL1=1.E-20
      NIT=100
      K=1
      U1=U4A
      C1=C4A
      UU = C1 * XS + U1
      RH1=RH4A
      E1=E4A
      P1=P4A
      GO TO (5, 25, 40), MQ
    5 CONTINUE
C     PRINT903
C 903 FORMAT(1H ,6X,"U2P",12X,"UUP",12X,"E2P",12X,"RH2P",11X,
C        "C2P",12X,
C    $"P2P")
      P2=P4B
      P2P = P4B
      IF((P2 - P1) .LE. 0.) P2 = P1 + 0.001
      U2 = U1 + (P2-P1)/(RH1*(UU-U1))
      RH2=(P2/P1)**1.4*RH1
   10 CONTINUE
      IF ( ABS(U2-U1) .LT. TOL1) GO TO 15
      U2P = (P2-P1)*(RH2-RH1)/(RH1*RH2*(U2-U1))+U1
```

```
   12 UUP=(P2-P1)/(RH1*(U2P-U1))+U1
   15 IF((RH2-RH1).LE.0.0)CALL EXIT
      E2P = E1 + (P1+P2)*(RH2-RH1)/(2.0*RH1*RH2)
      RH2P=EQSTRQ(L,E2P,P2)
      C2P=EQSTCQ(L,E2P,RH2P,P2)
      IF ( ABS ((U2P - U2 ) / U2P) -TOL ) 16, 16, 20
   16 IF (ABS(( RH2P - RH2 )/RH2P) - TOL ) 30,30,20
   20 IF ( K .GE. NIT ) GO TO 35
       IF(( K-(K/2)*2) .EQ. 0) GO TO 202
       RHXA=RH2
       RHYA=RH2P
       GO TO 205
  202  RHXB=RH2
       RHYB=RH2P
  205  IF (K .LT.12) GO TO 206
       IF (ABS(RHYB-RHYA).LT. 0.1E-06) GO TO 208
       BOB= (RHXB - RHXA)/(RHYB-RHYA)
       RH2= (RHXA-BOB*RHYA)/(1. -BOB)
       GO TO 208
  206  RH2=0.5*(RH2 + RH2P)
  208  K=K+1
       U2=0.5*(U2+U2P)
C     PRINT 900,U2P,UUP,E2P,RH2P,C2P,P2P
      GO TO 10
   35 PRINT 1000
      GO TO 30
   25 CONTINUE
      U2 = U4B
      U2P = U4B
      RH2 = RH1 * 1.1
   26 CONTINUE
      IF ( ABS (RH2 -RH1) .LT. TOL1) GO TO 27
      UUP = (RH2*U2 - RH1*U1)/(RH2-RH1)
   27 P2P = P1 + RH1*(UUP - U1)*(U2-U1)
      E2P = E1 + (P1+ P2P) * (RH2-RH1)/(2.*RH1*RH2)
      RH2P = EQSTRQ (L,E2P,P2P)
      C2P = EQSTCQ (L,E2P,RH2P,P2P)
        IF ( ABS((RH2P -RH2)/RH2P) .LT. TOL) GO TO 30
      K = K+1
      RH2 = (RH2P + RH2)/2.
C     PRINT 900,U2P,UUP,E2P,RH2P,C2P,P2P
      IF ( K .GE. NIT) GO TO 29
      GO TO 26
   29 PRINT 1000
      GO TO 30
   40 CONTINUE
      UUP = UU1
      RH2 = RH1 *2.0
      IF( ABS(EP) .LT. 0.001) RH2 = RH1
   41 U2P = (RH2*UUP - RH1*(UUP - U1))/RH2
```

```
      P2P = P1 + RH1*(UUP - U1)*(U2P - U1)
      E2P = E1 + (P1+ P2P) * (RH2-RH1)/(2.*RH1*RH2)
      RH2P = EQSTRQ (L,E2P,P2P)
      C2P = EQSTCQ (L,E2P,RH2P,P2P)
        IF ( ABS((RH2P -RH2)/RH2P) .LT. TOL) GO TO 30
      K = K+1
C     PRINT 900,U2P,UUP,E2P,RH2P,C2P,P2P
      IF ( K .GE. NIT) GO TO 39
      RH2 = (RH2P + RH2)/2.
      GO TO 41
   39 PRINT 1000
   30 CONTINUE
      U4B=U2P
      C4B=C2P
      RH4B=RH2P
      E4B=E2P
      P4B = P2P
 1999 FORMAT("       UUP=",E12.8)
      IF ( L .EQ. 1) UU1 = UUP
      IF ( L .EQ. 2) UU2 = UUP
      UUU = UUP
C     PRINT 900, U4B,UUP,E4B,RH4B,C4B,P4B
      RETURN
        END
      SUBROUTINE SWITCH(II,IADD,IMAT,ISKOS)
      COMMON/GAIN/Q(2,1000),X(2,1000),U(2,1000),C(2,1000),RH(2,10
     100),    E(2,1000),P(2,1000)
      COMMON/NOCON/IS1,IS2,IS3,IS4,INT1,INT2,MXNPT
      IF(IADD.LT.0) GO TO 1
      ISIGN=1
      N=0
      GO TO 2
 1      ISIGN=-1
      N=MXNPT-II
 2      CONTINUE
      ILAST=MXNPT+IADD-N-1
      DO 3 J=II,MXNPT
      IACH=MXNPT+IADD+(II-J)*ISIGN-N
      IOU=MXNPT+(II-J)*ISIGN-N
      X(1,IACH)=X(1,IOU)
      U(1,IACH)=U(1,IOU)
      C(1,IACH)=C (1,IOU)
      P(1,IACH)=P(1,IOU)
      RH(1,IACH)=RH(1,IOU)
      E(1,IACH)=E(1,IOU)
      Q(1,IACH)=Q(1,IOU)
 3      CONTINUE
      MXNPT=MXNPT+IADD
        RETURN
         END
```

DISTRIBUTION LIST

No. of Copies	Organization
12	Commander Defense Documentation Center ATTN: DDC-TCA Cameron Station Alexandria, Virginia 22314
2	Commander U.S. Army Materiel Command ATTN: AMCDMA Mr. N. Klein Mr. J. Bender 5001 Eisenhower Avenue Alexandria, Virginia 22333
1	Commander U.S. Army Materiel Command ATTN: AMCRD, BG H.A. Griffith 5001 Eisenhower Avenue Alexandria, Virginia 22333
1	Commander U.S. Army Materiel Command ATTN: AMCRD-T 5001 Eisenhower Avenue Alexandria, Virginia 22333
1	Commander U.S. Army Aviation Systems Command ATTN: AMSAV-E 12th and Spruce Streets St. Louis, Missouri 63166
1	Director U.S. Army Air Mobility Research and Development Laboratory Ames Research Center Moffett Field, California 94035
1	Commander U.S. Army Electronics Command ATTN: AMSEL-RD Ft. Monmouth, New Jersey 07703
1	Commander U.S. Army Missile Command ATTN: AMSMI-R Redstone Arsenal, Alabamb 35809
1	Commander U.S. Army Tank Automotive Command ATTN: AMSTA-RHFL Warren, Michigan 48090
2	Commander U.S. Army Mobility Equipment Research & Development Ctr. ATTN: Tech Docu Cen, Bldg. 315 AMSME-RZT Fort Belvoir, Virginia 22060
1	Commander U.S. Army Armament Command Rock Island, Illinois 61202
1	Commander U.S. Army Harry Diamond Labs. ATTN: AMXDO-TI 2800 Powder Mill Road Adelphi, Maryland 20783
1	Commander U.S. Army Materials and Mechanics Research Ctr. ATTN: AMXMR-ATL Watertown, Massachusetts 02172
1	Commander U.S. Army Natick Laboratories ATTN: AMXRE, Dr. D. Sieling Natick, Massachusetts 01762
3	HQDA (DARD-MS, Dr. W. Taylor; DARD-ARP; DARD-MD, Dr. P Friel) Washington, DC 20310

DISTRIBUTION LIST

No. of Copies	Organization
3	Commander U.S. Naval Air Systems Command ATTN: AIR-604 Washington, DC 20360
3	Commander U.S. Naval Ordnance Systems Command ATTN: ORD-9132 Washington, DC 20360
1	Commander U.S. Naval Weapons Center China Lake, California 93555
1	Director U.S. Naval Research Laboratory ATTN: Code 6240, Mr. Atkins Washington, DC 20390
1	AFATL (DLR) Eglin AFB Florida 32542
1	AFATL (DLRD) Eglin AFB Florida 32542
2	Commander U.S. Naval Ship Research and Development Center ATTN: D. R. Garrison A. Wilner Bethesda, Maryland 20034
1	AFATL (DLRV) Eglin AFB Florida 32542
1	AFWL Wright-Patterson AFB Ohio 45433

No. of Copies	Organization
1	Director Lawrence Livermore Laboratory ATTN: Mr. M. Wilkins P.O. Box 808 Livermore, California 94550
1	Director Los Alamos Scientific Lab P.O. Box 1663 Los Alamos, New Mexico 87544
1	Director National Aeronautics and Space Administration Langley Research Center Langley Station Hampton, Virginia 23365
1	Aerospace Corporation P.O. Box 95085 Los Angeles, California 90045
3	Drexel University ATTN: Prof. P.C. Chou Mechanics & Structures Group 32nd and Chestnut Streets Philadelphia, Pennsylvania 19137

<u>Aberdeen Proving Ground</u>

Marine Corps Ln Ofc
Dir, USAMSAA